STPC® Atlas
Programming Manual

Issue 1.0

July 2, 2002

STMicroelectronics

1 Table Of Contents

2 Liste of Tables

3 Liste of Figures

4. HOW TO USE THIS MANUAL

4.1. INTRODUCTION

This manual provides full technical documentation for the STPC device. It is recommended that the reader is familiar with the x86 series processors and PC compatible architectures before reading this document. Many terms are related directly to the PC architecture.

The manual itself is split into chapters. These chapters hold the information for a particular functional block of the device. For example, the chapter titled “Memory Access” gives the memory map of the STPC device, the memory architecture and interface to the external DRAM modules.

4.2. SPECIFIC NOTES

4.2.1. RESERVED BITS

Write mode 1 is a subset of Write Mode 0. No CPU-supplied write data is used. The read data latched from a previous read operation is written. The bit mask is disabled. The map-masks are implemented as they are for Write Mode 0.

Many bits in the register descriptions are noted as reserved. These bits are not internally connected, physically not present or are used for testing purposes. In all cases these bits should be set to a ‘0’ when writing to a register with reserved bits. When reading from a register with reserved bits, these specific bits should be masked from the data value before action is taken on the data.

Any functionality found by setting the reserved bits to levels other than ‘0’ cannot and will not be guaranteed on future revisions of the circuit design. Thus it is not recommended to use the bits marked as reserved in any way different from noted above.

4.2.2. SIGNAL ACTIVE STATE

The hash symbol (#) following a signal name indicates that when the signal is in its active (asserted) state, the signal is at a logic low level. When the “#” is not present at the end of a signal name, the logic high level represents the active state.

4.2.3. HEXADECIMAL NOTATION

In this manual Hexadecimal (Hex) numbers (numbers to the base 16: [0-9,A-F]) are denoted by the postfix ‘h’.

For example a memory address 783A hexadecimal will be written 783Ah.

4.2.4. ENDIAN

In common with the x86 architecture, values in memory are little-endian, that is the lower part of the memory contains the least significant Byte.

For an 8-bit value

N	7	6	5	4	3	2	1	0

For a 16-bit (word) value

N	7	6	5	4	3	2	1	0
N+1	15	14	13	12	11	10	9	8

For a 24-bit value

N	7	6	5	4	3	2	1	0
N+1	15	14	13	12	11	10	9	8
N+2	23	22	21	20	19	18	17	16

For a 32-bit (long word) value

N	7	6	5	4	3	2	1	0
N+1	15	14	13	12	11	10	9	8
N+2	23	22	21	20	19	18	17	16
N+3	31	30	29	28	27	26	25	24

For a 64-bit (QUAD word) value

N	7	6	5	4	3	2	1	0
N+1	15	14	13	12	11	10	9	8
N+2	23	22	21	20	19	18	17	16
N+3	31	30	29	28	27	26	25	24
N+4	39	38	37	36	35	34	33	32
N+5	47	46	45	44	43	42	41	40
N+6	55	54	53	52	51	50	49	48
N+7	63	62	61	60	59	58	57	56

4.3. ISSUING NOTES

There are three levels identified; Advanced data, Preliminary data and Full production release.

Each level is identified in a specific way as follows.

Document Identification	Status	Definition	Release Identification
ADVANCED DATA	In design	This document based on the product specification. The information may be updated without notice. Large changes may still occur.	Release A, Release B...
PRELIMINARY DATA	Pre-production Data	This document contains preliminary data and may be updated without notice in order to improve the product features.	Issue 0.X.
FULL PRODUCTION DATA	Production Data	This is the finalised document and all test plans are completed. The information may be updated without notice in order to improve the product features.	Issue 1.X.

5. LIST OF REGISTERS

This chapter lists all the registers accessable by software.

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
3.	**Power on strap Registers**			0022h	
3.1.1.	Strap Register 0	Strap0	Configuration	0023h	Index 04Ah
3.1.2.	Strap Register 1	Strap1	Configuration		Index 04Bh
3.1.3.	HCLK PLL Strap Register 0	HCLK_Strap0	Configuration		Index 05Fh
3.1.4.	Strap Register 2	Strap2	Configuration		Index 04Ch
6.	**Host Interface**				
6.5.	**Cache related registers**			0022h	
6.5.1.	Cache Architecture Register 0	Cash_Arc0	Configuration	0023h	Index 020h
6.5.2.	Cache Architecture Register 1	Cash_Arc1	Configuration		Index 021h
6.5.3.	Cache Architecture Register 2	Cash_Arc2	Configuration		Index 022h
6.6.	Address decode related registers			0022h	
6.6.1.	Memory Hole Control Register	MEM_HOLE	Configuration	0023h	Index 024h
6.6.2.	Shadow Control Register 0	SHADOW_0	Configuration		Index 025h
6.6.3.	Shadow Control Register 1	SHADOW_1	Configuration		Index 026h
6.6.4.	Shadow Control Register 2	SHADOW_2	Configuration		Index 027h
6.6.5.	Shadow Control Register 3	SHADOW_3	Configuration		Index 028h
6.7.	**Host DRAM controller registers**			0022h	
6.7.1.	SDRAM Bank 0 Register	SDRAM_Bank0	Configuration	0023h	Index 030h
6.7.2.	SDRAM Bank 1 Register	SDRAM_Bank1	Configuration		Index 031h
6.7.3.	SDRAM Bank 2 Register	SDRAM_Bank2	Configuration		Index 032h
6.7.4.	SDRAM Bank 3 Register	SDRAM_Bank3	Configuration		Index 033h
6.7.6.	SDRAM Refresh Register	SDRAM_Ref	Configuration		Index 039h
6.7.7.	Presents Detect Register	Pres_dect	Configuration		Index 097h
6.8.	**Watchdog Timer**				
6.8.4.	Watchdog Clock Divider register	WDCLKDVDRG	Configuration	0022h/0023h	0x00h
6.8.5.	Watchdog Count Load Register	WDCNTLDRG	Configuration		0x04h
6.8.6.	WatchDog Count Read Register	WDCNTRDRG	Configuration		0x08h
6.8.7.	WatchDog Control Register	WDCTRLREG	Configuration		0x0Ch
6.8.8.	WatchDog Counter Enable Register	WDCNTENREG	Configuration		0x10h
6.8.9.	Watchdog Status Register	WDSTREG	Configuration		0x14h
6.9.	**I/O Address Trapping**				
6.9.3.	I/O Address Registers	ADDREG0	Configuration		0x10h
6.9.3.	I/O Address Registers	ADDREG1	Configuration		0x14h
6.9.3.	I/O Address Registers	ADDREG2	Configuration		0x18h
6.9.3.	I/O Address Registers	ADDREG3	Configuration		0x1Ch
6.9.4.	IO Mask Registers	MASKREG0	Configuration		0x20h
6.9.4.	IO Mask Registers	MASKREG1	Configuration		0x24h
6.9.4.	IO Mask Registers	MASKREG2	Configuration		0x28h
6.9.4.	IO Mask Registers	MASKREG3	Configuration		0x2Ch
6.9.5.	Trapped address register	TADDREG	Configuration		0x30h

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
6.9.6.	Trapped Data Register	TDATAREG	Configuration		0x34h
6.9.7.	Trapped Byte Enable Register	TBEREG	Configuration		0x38h
6.9.8.	Control Register	CREG	Configuration		0x3Ch
6.9.9.	Trap Wait-for-Ready Generation Register	TWAITREG	Configuration		0x40h
6.9.10.	Status Register	STATREG	Configuration		0x44h
6.10.	**Real-Time 32-bit Memory Addressing**				
6.10.3.	Real Mode Aperture Register	RMAReg	Configuration		0x20h
6.10.4.	Remap Address Register	RAdrReg	Configuration		0x21h
6.10.5.	Control Register	CtlReg	Configuration		0x22h
7.3.	**Memory Interface**			GBase+4C6000h	
7.3.1.	Register 0	MEM_REG0	Configuration		Index 000h
7.3.2.	Register 1	MEM_REG1	Configuration		Index 004h
7.3.3.	Register 2	MEM_REG2	Configuration		Index 008h
7.4.	**MCLK Control Registers**			22h	
7.4.1.	MCLK Control Register 0	MCLK00		23h	Index 0x40h
7.4.2.	MCLK Control Register 1	MCLK01			Index 0x41h
8.5.	**North Bridge Configuration Address Registers**		Device = 0Bh	0xCF8h	IDSEL = ad[11]
8.3.	Configuration Address Register	Config_ Address	IO	0xCF8h	
8.4.	Configuration Data Register	Config_Data	IO	0xCFCh	
8.5.	North Bridge Vendor Identification Register	NB_V_ID	PCI Config		Index 0x0h
8.5.	North Bridge Device Identification Register	NB_D_ID	PCI Config		Index 0x2h
8.5.1.	North Bridge PCI Command Register	NB_Com	PCI Config		Index 0x4h
8.5.2.	North Bridge PCI Status Register	NB_Stat	PCI Config		Index 0x6h
8.5.3.	North Bridge PCI Revision Id Register	NB_R_ID	PCI Config		Index 0x8h
8.5.4.	North Bridge Device Class Code Register	NB_C_Code	PCI Config		Index 0x9h
8.5.5.	North Bridge Header Type Register	NB_Head	PCI Config		Index 0xEh
8.5.6.	North Bridge Control Register	NB_Cont	PCI Config		Index 0x50h
8.5.7.	North Bridge PCI Error Status Register	NB_E_Stat	PCI Config		Index 0x54h
8.7.	**South Bridge PCI Function 0 Configuration Registers**		Device=0Ch	0xCF8h	IDSEL = ad[12]
8.7.	South Bridge Vendor Identification Register	SB_V_ID0	PCI config	0xCFCh	Index 0x0h
8.7.	South Bridge Device Identification Register	SB_D_ID0	PCI Config		Index 0x2h
8.7.1.	South Bridge PCI Command Register	SB_Com_0	PCI Config		Index 0x4h
8.7.2.	South Bridge PCI Status Register	SB_Stat0	PCI Config		Index 0x6h
8.7.3.	South Bridge PCI Revision Id Register	SB_R_ID0	PCI Config		Index 0x8h
8.7.4.	South Bridge Device Class Code Register	SB_C_Code0	PCI Config		Index 0x9h

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
8.7.5.	South Bridge Header Type Register	SB_Head0	PCI Config		Index 0xEh
8.7.6.	South Bridge Miscellaneous Register	SB_Misc0			Index 040h
8.8.	**PCI to IDE Bridge Configuration Registers**		Device=0Dh	0xCF8h	IDSEL = ad[13]
8.8.1.	Vendor Identification Register	IDEB_V_ID1	PCI config F#1	0xCFCh	Index 0x0h
8.8.2.	Device Identification Register	IDEB_D_ ID1	PCI Config F#1		Index 0x2h
8.8.3.	PCI Command Register	IDEB_ Com1	PCI Config F#1		Index 0x4h
8.8.4.	PCI Status Register	IDEB_Stat1	PCI Config F#1		Index 0x6h
8.8.5.	Revision ID Register	IDEB_R_ID1	PCI Config F#1		Index 0x8h
8.8.6.	Programming Interface Register	Prog_Int	PCI Config F#1		Index 0x9h
8.8.7.	Sub-Class Code Register	Sub_Class	PCI Config F#1		Index 0xAh
8.8.8.	Base-Class code Register	Base_Class	PCI Config F#1		Index 0xBh
8.8.9.	Latency Timer control Register	Lat_T	PCI Config F#1		Index 0xDh
8.8.10.	Header Type Register	Head_T	PCI Config F#1		Index 0xEh
8.8.11.	IDE Base Address 0 Register	Base0	PCI Config F#1		Index 0x10h
8.8.12.	IDE Base Address 1 Register	Base1	PCI Config F#1		Index 0x14h
8.8.13.	IDE Base Address 2 Register	Base2	PCI Config F#1		Index 0x18h
8.8.14.	IDE Base Address 3 Register	Base3	PCI Config F#1		Index 0x1Ch
8.8.15.	IDE Base Address 4 Register	Base4	PCI Config F#1		Index 0x20h
8.8.16.	IDE Timing Register	IDE_Timing	PCI Config F#1		see Table 7-19
8.8.17.	IDE Miscellaneous Register	IDEB_Misc1	PCI Config F#1		Index 0x48h
8.9.	**PCI to USB Configuration Registers**		Device=0Eh	0xCF8h	IDSEL = ad[14]
8.9.1.	Vendor Identification Register	USBB_V_ID1	PCI Config F#1	0xCFCh	Index 0x0h
8.9.2.	Device Identification Register	USBB_D_ID1	PCI Config F#1		Index 0x2h
8.9.3.	PCI Command Register	USBB_Com	PCI Config F#1		Index 0x4h
8.9.4.	PCI Status Register	USBB_Stat	PCI Config F#1		Index 0x6h
8.9.5.	PCI Revision ID Register	USBB_R_ID	PCI Config F#1		Index 0x8h
8.9.6.	USB Class Code Register	USBB_C_Code	PCI Config F#1		Index 0x9h
8.9.7.	Header Type Register	USBB_Head	PCI Config F#1		Index 0xEh
8.10.	PCI Configuration for OpenHCI-compliant USB Host Controller				
8.10.1.	USB Command Register	Command	PCI Config F#1		Index
8.10.2.	Class Code Register	CLASS_CODE	PCI Config F#1		Index
8.10.3.	Base Address Register	BAR_OHCI	PCI Config F#1		Index
9.4.	**ISA standard Registers**				
9.4.1.	DMA 1 Channel 0 Base and Current Address	DMA1_CBA0	IO	0000h	
9.4.1.	DMA 1 Channel 0 Base and Current Count	DMA1_CBC0	IO	0001h	
9.4.1.	DMA 1 Channel 1 Base and Current Address	DMA1_CBA1	IO	0002h	

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
9.4.1.	DMA 1 Channel 1 Base and Current Count	DMA1_CBC1	IO	0003h	
9.4.1.	DMA 1 Channel 2 Base and Current Address	DMA1_CBA2	IO	0004h	
9.4.1.	DMA 1 Channel 2 Base and Current Count	DMA1_CBC2	IO	0005	
9.4.1.	DMA 1 Channel 3 Base and Current Address	DMA1_CBA3	IO	0006h	
9.4.1.	DMA 1 Channel 3 Base and Current Count	DMA1_CBC3	IO	0007h	
9.4.1.	DMA 1 Read Status / Write Command Register	DMA1_RSWC	IO	0008h	
9.4.1.	DMA 1 Request Register	DMA1_RR	IO	0009h	
9.4.1.	DMA 1 Read Command / Write Single Mask Register	DMA1_RCWSM	IO	000Ah	
9.4.1.	DMA 1 Mode Register	DMA1_Mode	IO	000Bh	
9.4.1.	DMA 1 Set / Clear Byte Pointer Flip - Flop	DMA1_SCBPFF	IO	000Ch	
9.4.1.	DMA 1 Read Temp Register / Master Clear	DMA1_RTMC	IO	000Dh	
9.4.1.	DMA 1 Clear Mask / Clear All Request	DMA1_CMCAR	IO	000Eh	
9.4.1.	DMA 1 Read / Write all Mask Register Bits	DMA1_RWMB	IO	000Fh	
9.4.2.	Interrupt Controller 1 Registers	IC_1	IO	0020h	
9.4.2.	Interrupt Controller 1 Mask Register	IC_1MR	IO	0021h	
9.4.3.	Interval Timer Register Counter 0 Count	IT_0	IO	0040h	
9.4.3.	Interval Timer Register Counter 1 Count	IT_1	IO	0041h	
9.4.3.	Interval Timer Register Counter 2 Count	IT_2	IO	0042h	
9.4.3.	Command Mode Register	IT_3	IO	0043h	
9.4.4.	Port Bh Register	Port _B	IO	0061h	
9.4.5.	Port 70h Register	Port_70	IO	0070h	
9.4.6.	Interrupt Controller 2 Registers	IC_2R	IO	00A0h	
9.4.6.	Interrupt Controller 2 Mask	IC_2M	IO	00A1h	
9.4.7.	DMA Controller 2 Registers	DMA_Cont2	IO		
9.4.7.	DMA 2 Channel 0 Base and Current Address	DMA2_CBA0	IO	00C0h	
9.4.7.	DMA2 Channel 0 Base and Current Count	DMA2_CBC0	IO	00C2h	
9.4.7.	DMA 2 Channel 1 Base and Current Address	DMA2_CBA1	IO	00C4h	
9.4.7.	DMA 2 Channel 1 Base and Current Count	DMA2_CBC1	IO	00C6h	
9.4.7.	DMA 2 Channel 2 Base and Current Address	DMA2_CBA2	IO	00C8h	
9.4.7.	DMA 2 Channel 2 Base and Current Count	DMA2_CBC2	IO	00CAh	
9.4.7.	DMA 2 Channel 3 Base and Current Address	DMA2_CBA3	IO	00CCh	

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
9.4.7.	DMA 2 Channel 3 Base and Current Count	DMA2_CBC3	IO	00CEh	
9.4.7.	DMA 2 Read Status / Write Command Register	DMA2_RSWC	IO	00D0h	
9.4.7.	DMA 2 Request Register	DMA2_RR	IO	00D2h	
9.4.7.	DMA 2 Read Command / Write Single Mask Register	DMA2_RCWSM	IO	00D4h	
9.4.7.	DMA 2 Mode Register	DMA2_Mode	IO	00D6h	
9.4.7.	DMA 2 Set / Clear Byte Pointer Flip - Flop	DMA2_SCBPFF	IO	00D8h	
9.4.7.	DMA 2 Read Temporary / Master Clear	DMA2_RTMC	IO	00DAh	
9.4.7.	DMA 2 Clear Mask / Clear All Requests Register	DMA2_CMCAR	IO	00DCh	
9.4.7.	DMA 2 Read / Write all Mask Register Bits	DMA2_RWMRB	IO	00DEh	
9.4.8.	DMA Page Registers	DMA_Page	IO		
9.4.8.	DMA Page Registers Port 80h (reserved)	Port_80	IO	0080h	
9.4.8.	DMA Page Register Channel 2	DMA_PRC2	IO	0081h	
9.4.8.	DMA Page Register Channel 3	DMA_PRC3	IO	0082h	
9.4.8.	DMA Page Register Channel 1	DMA_PRC1	IO	0082h	
9.4.8.	DMA Page Register Port 84h (Reserved)	Port_84	IO	0084h	
9.4.8.	DMA Page Register Port 85h (Reserved)	Port_85	IO	0085h	
9.4.8.	DMA Page Register Port 86h (Reserved)	Port_86	IO	0086h	
9.4.8.	DMA Page Register Channel 0	DMA_PRC0	IO	0087h	
9.4.8.	DMA Page Register Port 87h	Port_87	IO	0088h	
9.4.8.	DMA Page Register Channel 6	DMA_PRC6	IO	0089h	
9.4.8.	DMA Page Register Channel 7	DMA_PRC7	IO	008Ah	
9.4.8.	DMA Page Register Channel 5	DMA_PRC5	IO	008Bh	
9.4.8.	DMA Page Register Port 8Bh (Reserved)	Port_8B	IO	008Ch	
9.4.8.	DMA Page Register Port 8Ch (Reserved)	Port_8C	IO	008Dh	
9.4.8.	DMA Page Register Port 8Dh (Reserved)	Port_8D	IO	008Eh	
9.4.8.	DMA Page Register Port 8Eh (Reserved)	Port_8E	IO	008Fh	
9.5.	**ISA Configuration Registers**			0022h	
9.5.1.	Miscellaneous Control Register 0	Misc_Cont0	Configuration	0023h	Index 050h
9.5.2.	Miscellaneous Control Register 1	Misc_Cont1	Configuration		Index 051h
9.2.3.	PIRQA Routing control Register 0	PAR_Cont0	Configuration		Index 052h
9.6.	**Interrupt Router**				
9.6.2.	PIRQD Routing control Register 0	PDR_Cont0	Configuration		Index 055h
9.6.2.	PIRQC Routing control Register 0	PCR_Cont0	Configuration		Index 054h
9.6.2.	PIRQB Routing control Register 0	PBR_Cont0	Configuration		Index 053h
9.6.2.	PIRQA Routing control Register 0	PAR_Cont0	Configuration		Index 052h

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
9.6.2.	VMI IRQ Routing Control Register	VIR_Cont	Configuration		Index 058h
9.6.2.	IRQ Int Routing Control Register 5	IRQ_INT5	Configuration	0x410h	
9.6.2.	IRQ Int Routing Control Register 6	IRQ_INT6	Configuration	0x411h	
9.6.2.	IRQ Int Routing Control Register 7	IRQ_INT7	Configuration	0x412h	
9.6.2.	IRQ Int Routing Control Register 8	IRQ_INT8	Configuration	0x413h	
9.6.2.	IRQ Int Routing Control Register 9	IRQ_INT9	Configuration	0x414h	
9.6.2.	IRQ Int Routing Control Register 10	IRQ_INT10	Configuration	0x415h	
9.5.4.	Interrupt Level Control Register 0	IRQ_Lev_C_0	Configuration		Index 056h
9.5.5.	Interrupt Level Control Register 1	IRQ_Lev_C_1	Configuration		Index 057h
9.5.6.	IPC Configuration Register	IPC_Conf	Configuration		Index 001h
9.5.7.	VMI IRQ Routing Control Register	*VIR_Conf*	Configuration	022h/023h	Index 058h
9.5.8.	ISA Synchroniser Bypass Register	*ISA_Sync*	Configuration		Index 059h
9.6.2.	IRQ Ext Routing Control Register 15	IRQ_EXT15	Configuration	0x40Fh	
9.6.2.	IRQ Ext Routing Control Register 14	IRQ_EXT14	Configuration	0x40Eh	
9.6.2.	IRQ Ext Routing Control Register 13	IRQ_EXT13	Configuration	0x40Dh	
9.6.2.	IRQ Ext Routing Control Register 12	IRQ_EXT12	Configuration	0x40Ch	
9.6.2.	IRQ Ext Routing Control Register 11	IRQ_EXT11	Configuration	0x40Bh	
9.6.2.	IRQ Ext Routing Control Register 10	IRQ_EXT10	Configuration	0x40Ah	
9.6.2.	IRQ Ext Routing Control Register 9	IRQ_EXT9	Configuration	0x409h	
9.6.2.	IRQ Ext Routing Control Register 8	IRQ_EXT8	Configuration	0x408h	
9.6.2.	IRQ Ext Routing Control Register 7	IRQ_EXT7	Configuration	0x407h	
9.6.2.	IRQ Ext Routing Control Register 6	IRQ_EXT6	Configuration	0x406h	
9.6.2.	IRQ Ext Routing Control Register 5	IRQ_EXT5	Configuration	0x405h	
9.6.2.	IRQ Ext Routing Control Register 4	IRQ_EXT4	Configuration	0x404h	
9.6.2.	IRQ Ext Routing Control Register 3	IRQ_EXT3	Configuration	0x403h	
9.6.2.	IRQ Ext Routing Control Register 2	IRQ_EXT2	Configuration	0x402h	
9.6.2.	IRQ Ext Routing Control Register 1	IRQ_EXT1	Configuration	0x401h	
9.6.2.	IRQ Ext Routing Control Register 0	IRQ_EXT0	Configuration	0x400h	
9.7.	**DRQ Router**				
	Internal DRQ Routing Control Registers				
9.7.2.	IDRQ Internal Routing Control 0	IDRQ_cont0	Configuration	0x428h	
9.7.2.	IDRQ Internal Routing Control 1	IDRQ_cont1	Configuration	0x429h	
9.7.2.	IDRQ Internal Routing Control 2	IDRQ_cont2	Configuration	0x42Ah	
9.7.2.	IDRQ Internal Routing Control 3	IDRQ_cont3	Configuration	0x42Bh	
	External DRQ Routing Control Registers				
9.7.2.	EDRQ Internal Routing Control 0	EDRQ_cont0	Configuration	0x420h	
9.7.2.	EDRQ Internal Routing Control 1	EDRQ_cont0	Configuration	0x421h	
9.7.2.	EDRQ Internal Routing Control 2	EDRQ_cont0	Configuration	0x422h	
9.7.2.	EDRQ Internal Routing Control 3	EDRQ_cont0	Configuration	0x423h	
9.7.2.	EDRQ Internal Routing Control 4	EDRQ_cont0	Configuration	0x424h	
9.7.2.	EDRQ Internal Routing Control 5	EDRQ_cont0	Configuration	0x425h	
9.7.2.	EDRQ Internal Routing Control 6	EDRQ_cont0	Configuration	0x426h	
9.7.2.	EDRQ Internal Routing Control 7	EDRQ_cont0	Configuration	0x427h	
10.	**IDE Controller**		Configuration	See IDE Controller Chapter	

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
11.3.	**VGA Registers**				
11.4.	General Registers				
11.4.1.	Motherboard Enable Register	MBEN		0x094h	
11.4.2.	Add-in VGA Enable Register	ADDEN		0x46E8h	
11.4.3.	Video Subsystem Enable 1 Register	VSE1		0x102h	
11.4.4.	Video Subsystem Enable 2 Register	VSE2		0x3C3h	
11.4.5.	Miscellaneous Output Register	MISC		0x3CC/ 0x3C2h	
11.4.6.	Input Status Register #0	INP0		0x3C2h	
11.4.7.	Input Status Register #1	INP1		0x3XAh	
11.5.	Sequencer Registers				
11.5.1.	Sequencer Index Register	SRX		0x03C4h	
11.5.2.	Sequencer Reset Register	SR0		0x03C5h	Index 000h
11.5.3.	Sequencer Clocking Mode Register	SR1			Index 001h
11.5.4.	Sequencer Plane Mask Register	SR2			Index 002h
11.5.5.	Sequencer Character Map Register	SR3			Index 003h
11.5.6.	Sequencer Memory Mode Register	SR4			Index 004h
11.5.7.	Extended Register Lock/Unlock Register	SR6			Index 006h
11.6.	Graphics Controller Registers				
11.6.1.	Graphics Controller Index Register	GRX		0x03CEh	
11.6.2.	Graphics Set/Reset Register	GR0		0x03CFh	Index 000h
11.6.3.	Graphics Enable Set/Reset Register	GR1			Index 001h
11.6.4.	Graphics Colour Compare Register	GR2			Index 002h
11.6.5.	Raster Op/Rotate Count Register	GR3			Index 003h
11.6.6.	Graphics Read Map Select Register	GR4			Index 004h
11.6.7.	Graphics Mode Register	GR5			Index 005h
11.6.8.	Graphics Miscellaneous Register	GR6			Index 006h
11.6.9.	Graphics Colour Don't Care Register	GR7			Index 007h
11.6.10.	Graphics Bit Mask Register	GR8			Index 008h
11.7.	Attribute Controller Registers				
11.7.1.	Attribute Controller Index Register	ARX		0x3C0h	
11.7.2.	Attribute Palette Registers	AR0 - ARF		0x3C1/ 0x3C0h	
11.7.3.	Attribute Ctrl Mode Register	AR10		0x3C1/ 0x3C0h	
11.7.4.	Attribute Ctrl Overscan Colour Register	AR11		0x3C1/ 0x3C0h	
11.7.5.	Attribute Colour Plane Enable Register	AR12		0x3C1/ 0x3C0h	
11.7.6.	Attribute Horz Pixel Panning Register	AR13		0x3C1/ 0x3C0h	
11.7.7.	Attribute Colour Select Register	AR14		0x3C1/ 0x3C0h	
11.8.	**CRT Controller Registers**				
11.8.1.	Index Register	CRX	*see Note 1*	0x3X4h	
11.8.2.	Horizontal Total Register	CR0	*see Note 1*	0x3X5h	Index 000h
11.8.3.	Horiz display End Register	CR1			Index 001h

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
11.8.4.	Horiz Blanking Start Register	CR2			Index 002h
11.8.5.	Horiz Blanking End Register	CR3			Index 003h
11.8.6.	Horiz Retrace Start Register	CR4			Index 004h
11.8.7.	Horizontal Retrace End Register	CR5			Index 005h
11.8.8.	Vertical Total Register	CR6			Index 006h
11.8.9.	Overflow Register	CR7			Index 007h
11.8.10.	Screen A Preset Row Scan Register	CR8			Index 008h
11.8.11.	Character Cell Height Register	CR9			Index 009h
11.8.12.	Cursor Start Register	CRA			Index 00Ah
11.8.13.	Cursor End Register	CRB			Index 00Bh
11.8.14.	Start Address High Register	CRC			Index 00Ch
11.8.15.	Start Address Low Register	CRD			Index 00Dh
11.8.16.	Text Cursor Offset High Register	CRE			Index 00Eh
11.8.17.	Text Cursor Offset Low Register	CRF			Index 00Fh
11.8.18.	Vertical Retrace Start Register	CR10			Index 010h
11.8.19.	Vertical Retrace End Register	CR11			Index 011h
11.8.20.	Vertical Display End Register	CR12			Index 012h
11.8.21.	Offset Register	CR13			Index 013h
11.8.22.	Underline Location Register	CR14			Index 014h
11.8.23.	Vertical Blanking Start reg	CR15			Index 015h
11.8.24.	Vertical Blanking End Register	CR16			Index 016h
11.8.25.	Mode Register	CR17			Index 017h
11.8.26.	Line Compare Register	CR18			Index 018h
11.8.27.	Graphics Control Data	CR22			Index 022h
11.8.28.	Attribute Address Flip-flop	CR24			Index 024h
11.8.29.	Attribute Index Readback	CR26			Index 026h
11.9.	**VGA Extended Registers**		*see Note 1*	0x3X4h	
11.9.1.	Repaint Control Register 0	CR19	*see Note 1*	0x3X5h	Index 019h
11.9.2.	Repaint Control Register 1	CR1A			Index 01Ah
11.9.3.	Repaint Control Register 2	CR1B			Index 01Bh
11.9.4.	Repaint Control Register 3	CR1C			Index 01Ch
11.9.5.	Page Register 0	CR1D			Index 01Dh
11.9.6.	Page Register 1	CR1E			Index 01Eh
11.9.7.	Graphics Extended Enable Register	CR1F			Index 01Fh
11.9.8.	Graphics Extended GBASE Register	CR20			Index 020h
11.9.9.	Graphics Extended Aperture Register	CR21			Index 021h
11.9.10.	Repaint Control Register 4	CR25			Index 025h
11.9.11.	Repaint Control Register 5	CR27			Index 027h
11.9.12.	Palette Control Register	CR28			Index 028h
11.9.13.	Cursor Height Register	CR29			Index 029h
11.9.14.	Cursor Colour 0 Register A	CR2A			Index 02Ah
11.9.15.	Cursor Colour 0 Register B	CR2B			Index 02Bh
11.9.16.	Cursor Colour 0 Register C	CR2C			Index 02Ch
11.9.17.	Cursor Colour 1 Register A	CR2D			Index 02Dh
11.9.18.	Cursor Colour 1 Register B	CR2E			Index 02Eh
11.9.19.	Cursor Colour 1 Register C	CR2F			Index 02Fh
11.9.20.	Graphics Cursor Address Register 0	CR30			Index 030h

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
11.9.21.	Graphics Cursor Address Register 1	CR31			Index 031h
11.9.22.	Graphics Cursor Address Register 2	CR32			Index 032h
11.9.23.	Urgent Start Register	CR33			Index 033h
11.9.24.	Displayed Frame Y Offset 0 Register	CR34			Index 034h
11.9.25.	Displayed Frame Y Offset 1 Register	CR35			Index 035h
11.9.26.	Interlace Half Field Start Register	CR39			Index 039h
11.9.27.	Implementation Number Register	CR3A			Index 03Ah
11.9.28.	Graphics Version Register	CR3B			Index 03Bh
		CR3C			Index 03Ch
		CR3D			Index 03Dh
11.9.29.	Miscellaneous Test Register	CR3E			Index 03Eh
11.9.30.	DDC Control Register	CR3F			Index 03Fh
11.9.31.	TV Interface Control Register	CR40			Index 040h
11.9.32.	TV Horizontal Active Video Start A Register	CR41			Index 041h
11.9.33.	TV Horizontal Active Video Start B Register	CR42			Index 042h
11.9.34.	TV Horizontal Sync End A Register	CR43			Index 043h
11.9.35.	TV Horizontal Sync End B Register	CR44			Index 044h
11.12.	**RAMDAC Registers**				
11.12.1.	Palette Pixel Mask Register	Pixel_Mask		0x3C6h	
11.12.2.	Palette Read Index Register	Read_Index		0x3C7h	
11.12.3.	Palette State Register	Palette_State		0x3C7h	
11.12.4.	Palette Write Index Register	Write_Index		0x3C8h	
11.12.5.	Palette Data Register	Palette_Data		0x3C9h	
11.13	**DCLK Control Register**			022h	
11.13.1.	DCLK Control Register 00	DCLK00	PCI Config	023h	Index 0x42h
11.13.2.	DCLK Control Register 01	DCLK01	PCI Config		Index 0x43h
11.13.3.	DCLK Control Register 10	DCLK10	PCI Config		Index 0x44h
11.13.4.	DCLK Control Register 11	DCLK11	PCI Config		Index 0x45h
11.13.5.	DCLK Control Register 20	DCLK20	PCI Config		Index 0x46h
11.13.6.	DCLK Control Register 21	DCLK21	PCI Config		Index 0x47h
11.13.7.	DCLK Control Register 30	DCLK30	PCI Config		Index 0x48h
11.13.8.	DCLK Control Register 31	DCLK31	PCI Config		Index 0x49h
12.	**Graphics Engine**				
12.5.	**VGA Operand Sources**				
12.9.1.	Back Ground Colour Register	Background		8400000h	Index 0x004h
12.9.2.	Cursor Coordinate Register	Cursor_XY			Index 0x11Ch
12.9.3.	Top of Data FIFO Register	Data_Port			Index 0x804h
12.9.4.	Destination Operand Base Address Register	Dst_Base			Index 0x018h
12.9.5.	Destination Pitch Register	Dst_Pitch			Index 0x028h

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
12.9.6.	Destination Operand Coordinate Register	Dst_XY		8410000h	
12.9.7.	Foreground Colour Register	Foreground		8400000h	Index 0x034h
12.9.8.	Height Register	Height			Index 0x048h
12.9.9.	Pattern Base Address Operand Register	Pattern			Index 0x058h
12.9.10.	Pixel Depth Operand Register	Pixel_Depth			Index 0x07Ch
12.9.11.	Raster Operation Register	ROP			Index 0x08Ch
12.9.12.	Source Base Address Operand Register	Src_Base			Index 0x098h
12.9.13.	Source Pitch Operand Register	Src_Pitch			Index 0x0ACh
12.9.14.	Source Coordinate Register	Src_XY			Index 0x0BDh
12.9.15.	Status Register	Status			Index 0x908
12.9.16.	Width Register	Width			Index 0x0C8h
12.9.17.	Extra Use Register	Xtra			Index 0x0D4h
12.9.18.	SRC Transparency Compare Register	SRC_ Transparency			Index 0xECh
12.9.19.	DST Transparency Compare Register	DST_ Transparency			Index 0xFCh
13.	**Line Draw Engine**			GBASE+ 4D5000h	
13.5.1.	XStart Register	XSTART_REG			Index 0x00h
13.5.2.	YStart Register	YSTART_REG			Index 0x04h
13.5.3.	XEnd Register	XEND_REG			Index 0x08h
13.5.4.	YEnd Register	YEND_REG			Index 0x0Ch
13.5.5.	DX Register	DX_REG			Index 0x10h
13.5.6.	DY Register	DY_REG			Index 0x14h
13.5.7.	Scaled DX Register	SCALEDDX_ REG			Index 0x18h
13.5.8.	Scaled DY Register	SCALEDDY_ REG			Index 0x1Ch
13.5.9.	FXY Register	FXY_REG			Index 0x20h
13.5.10.	Scaled FXY Register	SCALEDFXY_ REG			Index 0x24h
13.5.11.	End Point Correct Register	CORRECT_REG			Index 0x28h
13.5.12.	LUT Register	LUT_REG			Index 0x2Ch
13.5.13.	Line Width Register	LINEWIDTH_ REG			Index 0x30h
13.5.14.	Line Colour Register	LINECOLOUR_ REG			Index 0x34h
13.5.15.	Black Colour Register	BLACKCOLOUR _REG			Index 0x38h
13.5.16.	Misc Register	MISC_REG			Index 0x3Ch
13.5.17.	FBBase Address Register	FBBASEADDR			Index 0x40h
13.5.18.	Frame Buffer Shifts Register	FBPITCHMULTI _REG			Index 0x44h
13.5.19.	Frame Buffer Pitch Register	FBPITCH_REG			Index 0x48h
13.5.20.	Max Clip Coordinates Register	CLIPMAXCORDI _REG			Index 0x4Ch

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
13.5.21.	Command Register	COMMAND_ REG			Index 0x50h
13.5.22.	Status Register	STATUS_REG			Index 0x54h
13.5.23.	Software Reset Register	SOFT_RESET_ REG			Index 0x58h
13.5.24.	Performance Register	PERFORM- ANCEREG			Index 0x5Ch
13.5.25.	Flush Count Register	FLUSH_REG			Index 0x60h
13.5.26.	Min Clip Coordinates Register	CLIPMINCORDI _REG			Index 0x64h
14.	**Alpha Font Engine**			GBASE + 4D4000h	
14.7.1.	Background Colour Register	BACKGROUND			Index 0x00h
14.7.2.	Data Port Register	DATAPORT			Index 0x04h
14.7.3.	Destination Base Address Register	DST_BASE			Index 0x08h
14.7.4.	Destination Pitch Register	DST_PITCH			Index 0x0Ch
14.7.5.	Destination Coordinates Register	DST_XY			Index 0x10h
14.7.6.	Foreground Colour Register	FOREGROUND			Index 0x14h
14.7.7.	Font Register	FONT			Index 0x18h
14.7.8.	Status Register	STATUS			Index 0x1Ch
14.7.9.	Source Coordinates Register	SRC_XY			Index 0x20h
14.7.10.	Source Base Address Register	SRC_BASE			Index 0x24h
14.7.11.	Source Pitch Register	SRC_PITCH			Index 0x28h
14.7.12.	ROP	ROP			Index 0x2Ch
14.7.13.	Source Water Mark Register	SRC_WATER_ MARK			Index 0x30h
14.7.14.	Destination Water Mark Register	DST_WATER_ MARK			Index 0x34h
14.7.15.	Data Drain Water Mark Register	DST_WATER_ MARK			Index 0x38h
14.7.16.	Add Offset Register	ADD_OFFSET			Index 0x3Ch
14.7.17.	Misc Register	MISC			Index 0x40h
14.7.18.	Software Reset Register	SOFT_RESET			Index 0x44h
14.7.19.	Current Performance Count Register	CUR_PERF_ REG			Index 0x48h
14.7.20.	Previous Performance Count Register	PVS_PERF_ REG			Index 0x4Ch
14.7.21.	Command Font Register	CMD_FONT_ REG			Index 0x50h
14.7.22.	Command Dst_XY Register	CMD_DSTXY_ REG			Index 0x54h
14.7.23.	Command Dst_Base Register	CMD_DSTBASE _REG			Index 0x58h
14.7.24.	Command Debug Register	CMD_DEBUG_ REG			Index 0x5Ch
15.	**Video Input Port**				

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
15.6.	**Video Input Port Registers**				
15.6.1.	Frame Buffer Address Readback	FB1_Adr			Index 0x00h
15.6.2.	Video Inport Port Configuration Register	vin_CFG			Index 0x04h
15.6.3.	Video Input Port Status Register	vin_stat			Index 0x08h
15.6.4.	Video Input Buffer Address0	vin_ad0			Index 0x0Ch
15.6.5.	Video Input Buffer Address1	vin_ad1			Index 0x10h
15.6.6.	Video Input Destination Pitch	vin_dp			Index 0x14h
15.6.7.	External Timing Generator1	vtg_ext1			Index 0x28h
15.6.8.	External Timing Generator2	vtg_ext2			Index 0x2Ch
15.6.9.	Horizontal Timing Generator	vtg_ht			Index 0x30h
15.6.10.	Video Timing Generator	vtg_vt			Index 0x34h
16.	**Video Pipeline Registers**			x480000h	
16.3.1.	Video Source Base Register	Video_Src_ Base			Index 0x00h
16.3.2.	Video Source Pitch Register	Video_Src_ Pitch			Index 0x04h
16.3.3.	Video Source Dimension Register	Video_Src_ Dim			Index 0x08h
16.3.4.	CRTC Burst Length Register	CRTC_Burst_ Length			Index 0x0Ch
16.3.5.	Video Burst Length Register	Video_Burst_ Length			Index 0x10h
16.3.6.	**Destination Specification Registers**				
16.3.6.1.	Video Destination XY Register	Video_Dst_XY			Index 0x14h
16.3.6.2.	Video Destination Dimension Register	Vid_Dst_ Dim			Index 0x18h
16.4.	**Filter Control Registers**				
16.4.1.	Horizontal Scaling and Decimation Register	Horiz_Scale			Index 0x20h
16.4.2.	Vertical Scaling and Decimation Register	Vert_Scale			Index 0x28h
16.4.3.	Colour Space Converter Specification Register	Clr_Con_Spec			Index 0x2Ch
16.5.	**Video and Graphics Mixing Control Registers**				
16.5.1.	Mix Mode Register	Mix_Mode			Index 0x30h
16.5.2.	Colour Key Register	CLR_Key			Index 0x34h
16.5.3.	Chroma Key Low Register	CKL			Index 0x38h
16.5.4.	Chroma Key High Register	CKH			Index 0x3Ch
16.5.5.	Status Register	Filter_Stat			Index 0x40h
17.3.	**TFT Configuration Registers**				
17.4.1.	Active Input Pixel Count	pxlPerInline	Configuration	0x084CC000	
17.4.1.	Input Back Porch	hsync2Dsp1En	Configuration		

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
17.4.2.	FP Horizontal Sync Width	fplinePlsWdth	Configuration	0x084CC004	
17.4.2.	FP Horizontal Back Porch	hrzntlBckPrch	Configuration		
17.4.3.	FP Active Pixel Count	pxlPerOutLine	Configuration	0x084CC008	
17.4.3.	FP Horizontal Front Porch	hrzntlFrntPrch	Configuration		
17.4.4.	FP Vertical Sync Width	framePlsWdth	Configuration	0x084CC00C	
17.4.4.	FP Vertical Back Porch	vrtclBckPrch	Configuration		
17.4.5.	FP Active Line Count	totalActvLines	Configuration	0x084CC010	
17.4.5.	Interface Control	InterfaceCon-trol	Configuration		
17.4.6.	Blank red	PWM_Cont	Configuration	0x084CC014	
	Blank green	PWR_Cont	Configuration		
	Blank blue	B_Red	Configuration		
	Power control	B_Green	Configuration		
	Polarity control	B_Blue	Configuration		
17.4.7.	PWM Control	Pol_Cont	Configuration	0x084CC018	
	PCMCIA Registers				
18.9.	General Setup Registers			0xCF8h/ 0xCFCh	
18.9.1.	Identification and Revision	Id_Rev	Configuration		Index 0x00h
18.9.2.	Interface Status	Int_Stat	Configuration		Index 0x01h
18.9.3.	Power and RESETDRV Control	PRD_Cont	Configuration		Index 0x02h
18.9.4.	Card Status Change	CSC	Configuration		Index 0x04h
18.9.5.	Address Window Enable	AW_En	Configuration		Index 0x06h
18.9.6.	Card Detect and General Control	C_Cont	Configuration		Index 0x16h
18.9.7.	Global Control Register	Global	Configuration		Index 0x1Eh
18.10.	**Interrupt Register**				
18.10.1.	Interupt and General Control	Int_Cont			Index 0x03h
18.10.2.	Card Status Change Interupt Configu-ration	CSCIC			Index 0x05h
18.11.	**IO Registers**				
18.11.1.	I/O Control	I/O_Cont			Index 0x07h
18.11.2.	I/O Address 0 Start Low Byte	Add0_SLB			Index 0x08h
18.11.3.	I/O Address 0 Start High Byte	Add0_SHB			Index 0x09h
18.11.4.	I/O Address 0 Stop Low Byte	Add0_SLB			Index 0x0Ah
18.11.5.	I/O Address 0 Stop High Byte	Add0_SHB			Index 0x0Bh
18.11.6.	I/O Address 1 Start Low Byte	Add2_SLB			Index 0x0Ch
18.11.7.	I/O Address 1 Start High Byte	Add2_SHB			Index 0x0Dh
18.11.8.	I/O Address 1 Stop Low Byte	Add2_StLB			Index 0x0Eh
18.11.9.	I/O Address 1 Stop High Byte	Add2_StHB			Index 0x0Fh
18.12.	**Memory Registers**				
18.12.1.	System Memory Address 0 Mapping Start Low Byte	Mem_Add0_SLB			Index 0x10h
18.12.2.	System Memory Address 0 Mapping Start High Byte	Mem_Add0_SH B			Index 0x11h

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
18.12.3.	System Memory Address 0 Mapping Stop Low Byte	Mem_Add0_StLB			Index 0x12h
18.12.4.	System Memory Address0 Mapping Stop High Byte	Mem_Add0_StHB			Index 0x13h
18.12.5.	Card Memory Offset Address 0 Low Byte	CM_Add0_LB			Index 0x14h
18.12.6.	Card Memory Offset Address 0 High Byte	CM_Add0_HB			Index 0x15h
18.12.7.	System Memory Address 1 Mapping Start Low Byte	SM_Add1_MSLB			Index 0x18h
18.12.7.	System Memory Address 1 Mapping Start High Byte	SM_Add1_MSHB			Index 0x19h
18.12.7.	System Memory Address 1 Mapping Stop Low Byte	SM_Add1_MStLB			Index 0x1Ah
18.12.7.	System Memory Address 1 Mapping Stop High Byte	SM_Add1_MStHB			Index 0x1Bh
18.12.7.	Card Memory Offset Address 1 Low Byte	CM_Add1_LB			Index 0x1Ch
18.12.7.	Card Memory Offset Address 1 High Byte	CM_Add1_HB			Index 0x1Dh
18.12.7.	System Memory Address 2 Mapping Start Low Byte	SM_Add2_MSLB			Index 0x20h
18.12.7.	System Memory Address 2 Mapping Start High Byte	SM_Add2_MSHB			Index 0x21h
18.12.7.	System Memory Address 2 Mapping Stop Low Byte	SM_Add2_MStLB			Index 0x22h
18.12.7.	System Memory Address 2 Mapping Stop High Byte	SM_Add2_MStHB			Index 0x23h
18.12.7.	Card Memory Offset Address 2 Low Byte	CM_Add2_LB			Index 0x24h
18.12.7.	Card Memory Offset Address 2 High Byte	CM_Add2_HB			Index 0x25h
18.12.7.	System Memory Address 3 Mapping Start Low Byte	SM_Add3_MSLB			Index 0x28h
18.12.7.	System Memory Address 3 Mapping Start High Byte	SM_Add3_MPHB			Index 0x29h
18.12.7.	System Memory Address 3 Mapping Stop Low Byte	SM_Add3_MPStLB			Index 0x2Ah
18.12.7.	System Memory Address 3 Mapping Stop High Byte	SM_Add3_MStHB			Index 0x2Bh
18.12.7.	Card Memory Offset Address 3 Low Byte	CM_Add3_LB			Index 0x2Ch
18.12.7.	Card Memory Offset Address 3 High Byte	CM_Add3_HB			Index 0x2Dh
18.12.7.	System Memory Address 4 Mapping Start Low Byte	SM_Add4_MSLB			Index 0x30h
18.12.7.	System Memory Address 4 Mapping Start High Byte	SM_Add4_MSHB			Index 0x31h
18.12.7.	System Memory Address 4 Mapping Stop Low Byte	SM_Add4_MStLB			Index 0x32h
18.12.7.	System Memory Address 4 Mapping Stop High Byte	SM_Add4_MStHB			Index 0x33h
18.12.7.	Card Memory Offset Address 4 Low Byte	CM_Add4_LB			Index 0x34h
18.12.7.	Card Memory Offset Address 4 High Byte	CM_Add4_HB			Index 0x35h

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
19.4.	**Keyboard/Mouse Controller Registers**				
19.4.1.	Input Buffer Data Register	In_Buf	I/O	60h	
19.4.2.	Output Buffer Register	Out_Buf	I/O	60h	
19.4.3.	Command Byte Register	Com_Byte	I/O	60h	
19.4.4.	Command Register	Com_Reg	I/O	64h	
19.4.5.	Status Register	Stat_Reg	I/O	64h	
20.4.	**Local Bus Configuration Registers**		16 bit access	Variable	
20.5.4.	I/O Slot Base Register 0	IOAREG0			00h
20.5.4.	I/O Slot Base Register 1	IOAREG1			02h
20.5.4.	I/O Slot Base Register 2	IOAREG2			04h
20.5.4.	I/O Slot Base Register 3	IOAREG3			06h
20.5.4.	I/O Slot Base Register 4	IOAREG4			08h
20.5.4.	I/O Slot Base Register 5	IOAREG5			0Ah
20.5.4.	I/O Slot Base Register 6	IOAREG6			0Ch
20.5.4.	I/O Slot Base Register 7	IOAREG7			0Eh
20.5.5.	I/O Slot Mask Register 0	IOMREG0			10h
20.5.5.	I/O Slot Mask Register 1	IOMREG1			12h
20.5.5.	I/O Slot Mask Register21	IOMREG2			14h
20.5.5.	I/O Slot Mask Register 3	IOMREG3			16h
20.5.5.	I/O Slot Mask Register 4	IOMREG4			18h
20.5.5.	I/O Slot Mask Register 5	IOMREG5			1Ah
20.5.5.	I/O Slot Mask Register 6	IOMREG6			1Ch
20.5.5.	I/O Slot Mask Register 7	IOMREG7			1Eh
20.5.6.	Memory Base Address Register 0	MEMAREG0			38h
20.5.7.	Memory Base Address Register 1	MEMAREG1			3Ah
20.5.8.	Memory Mask Register	MEMMASK			3Ch
20.6.	**Local Bus Timing Registers**				
20.6.1.	Memory Timing Template 0	TIMEMEM0			20h
20.6.2.	Memory Timing Template 1	TIMEMEM1			22h
20.6.3.	I/O Timing Template 0	TIMEIO0			24h
20.6.4.	I/O Timing Template 1	TIMEIO1			26h
20.6.5.	I/O Timing Template 2	TIMEIO2			28h
20.6.6.	I/O Timing Template 3	TIMEIO3			2Ah
20.6.7.	I/O Timing Template 4	TIMEIO4			2Ch
20.6.8.	I/O Timing Template 5	TIMEIO5			2Eh
20.6.9.	I/O Timing Template 6	TIMEIO6			30h
20.6.10.	I/O Timing Template 7	TIMEIO7			32h
20.7.	**Local Bus Control Register**				
20.7.	Control Register	CONTROL			34h
20.8.	I/O Width Register	IOWIDTH			36h
	GPIO Interface Registers			0320h/0328h	

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
21.3.1.	Port Direction Control Register	portDirCtrlReg			Index 0x0h
21.3.2.	Read Port Control Register	readPortCtrl			Index 0x1h
21.3.3.	Read Register	readReg			Index 0x2h
21.3.4.	Interrupt Unmask Register	intrUnMaskReg			Index 0x3h
21.3.5.	Interrupt Edge Register	intrEdgeSelect			Index 0x4h
21.3.6.	Interrupt Clear Command Register	ClearIntr			Index 0x5h
21.3.7.	GPIO Port Register	GPIOport			Index 0x6h
21.3.8.	Strap Register	StrapReg			Index 0x7h
	Universal Serial Bus				
22.2.	Operational Registers				
22.3.	PCI Configuration				
	Serial Port			03F8h/02F8h	
23.4.2.	Receiver Buffer Register	RBR			Index 0x00h
23.4.3.	Transmitter Holding Register	THR			Index 0x00h
23.4.4.	Interrupt Enable Register	IER			Index 0x01h
23.4.5.	Interrupt Identification Register	IIR			Index 0x02h
23.4.9.	FIFO Control Register	FCR			Index 0x02h
23.4.10.	Line Control Register	LCR			Index 0x03h
23.4.11.	Modem Control Register	MCR			Index 0x04h
23.4.12.	Line Status Register	LSR			Index 0x05h
23.4.13.	Modem Status Register	MSR			Index 0x06h
23.4.14.	Scratch Register				Index 0x07h
23.4.15.	Divisor Latch (LS) Register	DLL			Index 0x00h
23.4.15.	Divisor Latch (MS) Register	DLM			Index 0x01h
	Parallel Port				
24.3.1.2.	Configuration Select Register	CSR		03F0h	Index 0x00h
24.3.1.3.	Configuration Register 1	CR1		03F1h	Index 0x01h
24.3.1.4.	Configuration Register 4	CR4			Index 0x04h
24.3.2.2.	Status Register	P_Stat		0278h/0378h	Index 0x01h
24.3.2.3.	Control Register	P_CTRL			Index 0x02h
25.2.	**Power Management Controller Registers**			0022h	
25.2.1.	Timer Register 0	Timer0	Configuration	0023h	Index 060h
25.2.2.	Timer Register 1	Timer1	Configuration		Index 061h
25.2.3.	Timer Register 2	Timer2	Configuration		Index 08Dh
25.2.4.	System Activity Enable Register 0	Sys_Activ_en0	Configuration		Index 062h
25.2.5.	System Activity Enable Register 1	Sys_Activ_en1	Configuration		Index 063h
25.2.6.	System Activity Enable Register 2	Sys_Activ_en2	Configuration		Index 064h
25.2.7.	House-Keeping Activity Enable Register 0	HK_Activ_en0	Configuration		Index 065h
25.2.8.	House-Keeping Activity Enable Register 1	HK_Activ_en1	Configuration		Index 066h
25.2.9.	Peripheral Inactivity Detection Register 0	Perif_Inact0	Configuration		Index 067h
25.2.10.	Peripheral Activity Detection Register 0	Perif_Act0	Configuration		Index 069h

Table 5-1. Registers Described in this Manual

Section	Register Name	Mnemonic	Purpose	Address	Access type
25.2.11.	Peripheral Activity Detection Register 1	Perif_Act1	Configuration		Index 06Ah
25.2.12.	Address Range 0 Register 0	Add_ Rang0-0	Configuration		Index 06Bh
25.2.13.	Address Range 0 Register 1	Add_ Rang0-1	Configuration		Index 06Ch
25.2.14.	SMI Control Register 0	SMI_Cont0	Configuration		Index 071h
25.2.15.	SMI Status Register 0	SMI_Stat0	Configuration		Index 073h
25.2.16.	SMI Status Register 1	SMI_Stat1	Configuration		Index 074h
25.2.17.	Peripheral Inactivity Status Register 0	Perif_Stat0	Configuration		Index 075h
25.2.18.	Activity Status Register 0	Activ_Stat0	Configuration		Index 077h
25.2.19.	Activity Status Register 1	Activ_Stat1	Configuration		Index 078h
25.2.20.	Activity Status Register 2	Activ_Stat2	Configuration		Index 079h
25.2.21.	PMU State Register	PMU	Configuration		Index 07Ah
25.2.22.	General Purpose Register	GP	Configuration		Index 07Bh
25.2.23.	Clock Control Register 0	Clk_Cont0	Configuration		Index 07Ch
25.2.24.	Doze Timer Read Back Register	Doze	Configuration		Index 088h
25.2.25.	Standby Timer Read Back Register	Standby	Configuration		Index 089h
25.2.26.	Suspend Timer Read Back Register	Suspend	Configuration		Index 08Ah
25.2.27.	House-Keeping Timer Read Back Register	HK_Timer	Configuration		Index 08Bh
25.2.28.	Peripheral Timer Read Back Register	Perif_Timer	Configuration		Index 08Ch
Note 1: X can stand for B (Monochrome Display) or D (Color Display)					

LIST OF REGISTERS

These registers are described in the ST 486 Datasheet.

Section	Register Name	Mnemonic	Purpose	Address	Access type
2.4.4.3	**Configuration Registers**			22h	
2.4.4.3	Configuration Control 1	CCR1	IO	23h	C1h
2.4.4.3	Configuration Control 2	CCR2	IO		C2h
2.4.4.3	Configuration Contol 3	CCR3	IO		C3h
2.4.4.3	SMM Address Region	SMAR	IO		CDH
2.4.4.3	Device Indentification 0	DIR0	IO		FEh
2.4.4.3	Device Identification 1	DIR1	IO		FFh

Table 5-2. CPU Registers located in the ST 486 Manual

6. HOST INTERFACE

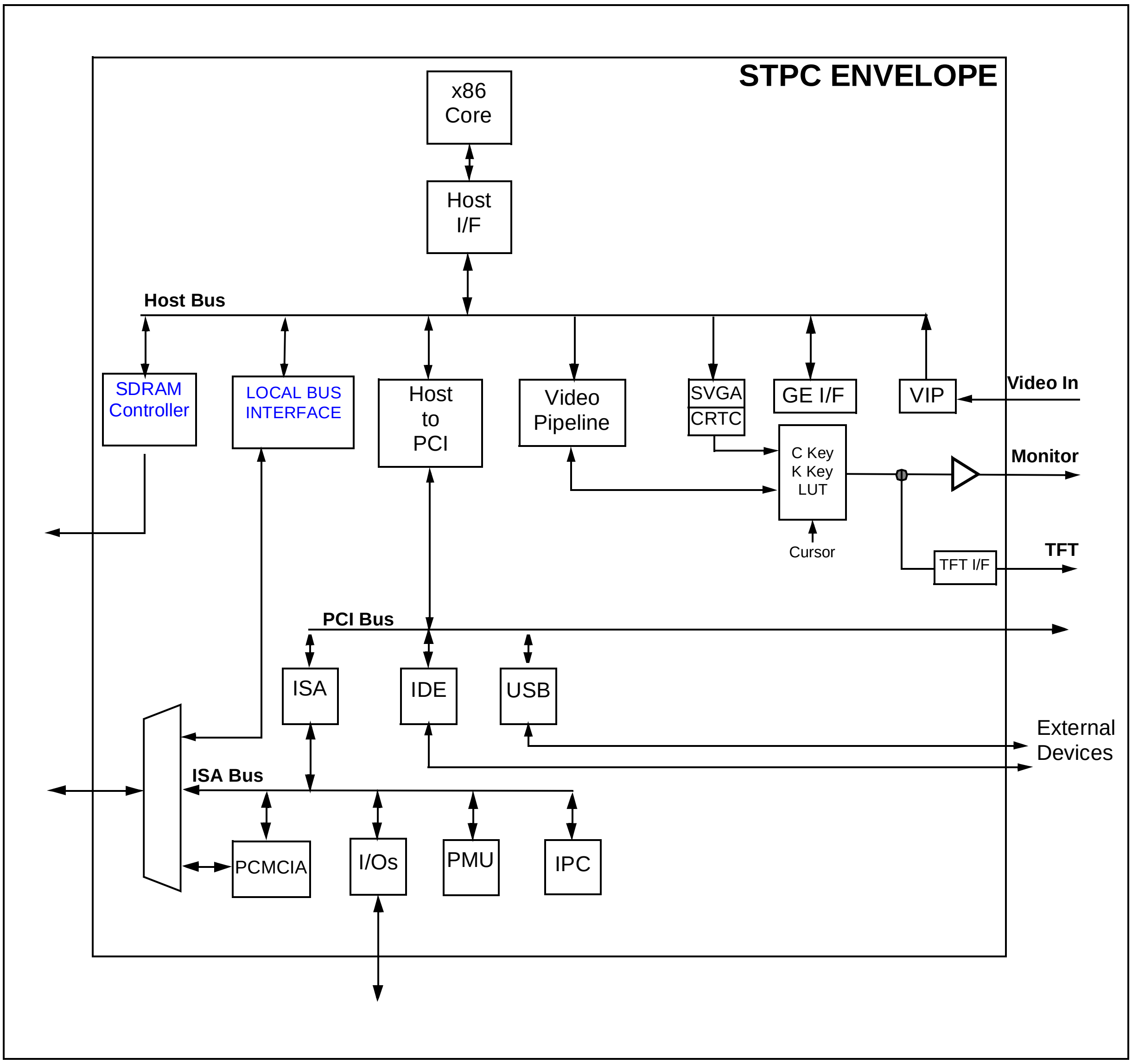
6.1. INTRODUCTION

This Chapter describes the Memory and I/O Mapping of the STPC with details on how to configure the Cache Memory registers.

This Chapter also describes the three additional functions of the Host Module, the interface between the CPU and peripheral devices. These functions comprise the a Watchdog Timer (Section 6.8.), I/O Address Trapping (Section 6.9.) and Real-Mode 32-bit Memory Addressing (Section 6.10.). The final section (Section 6.11.), Accessing Configuration Registers, is applicable to all three functions.

The Host is the main interface between the CPU and the other integrated peripherals of the STPC. Figure 6-1 below illustrates the relation of the integrated devices with reference to the Host Interface.

Figure 6-1. STPC Host Layout



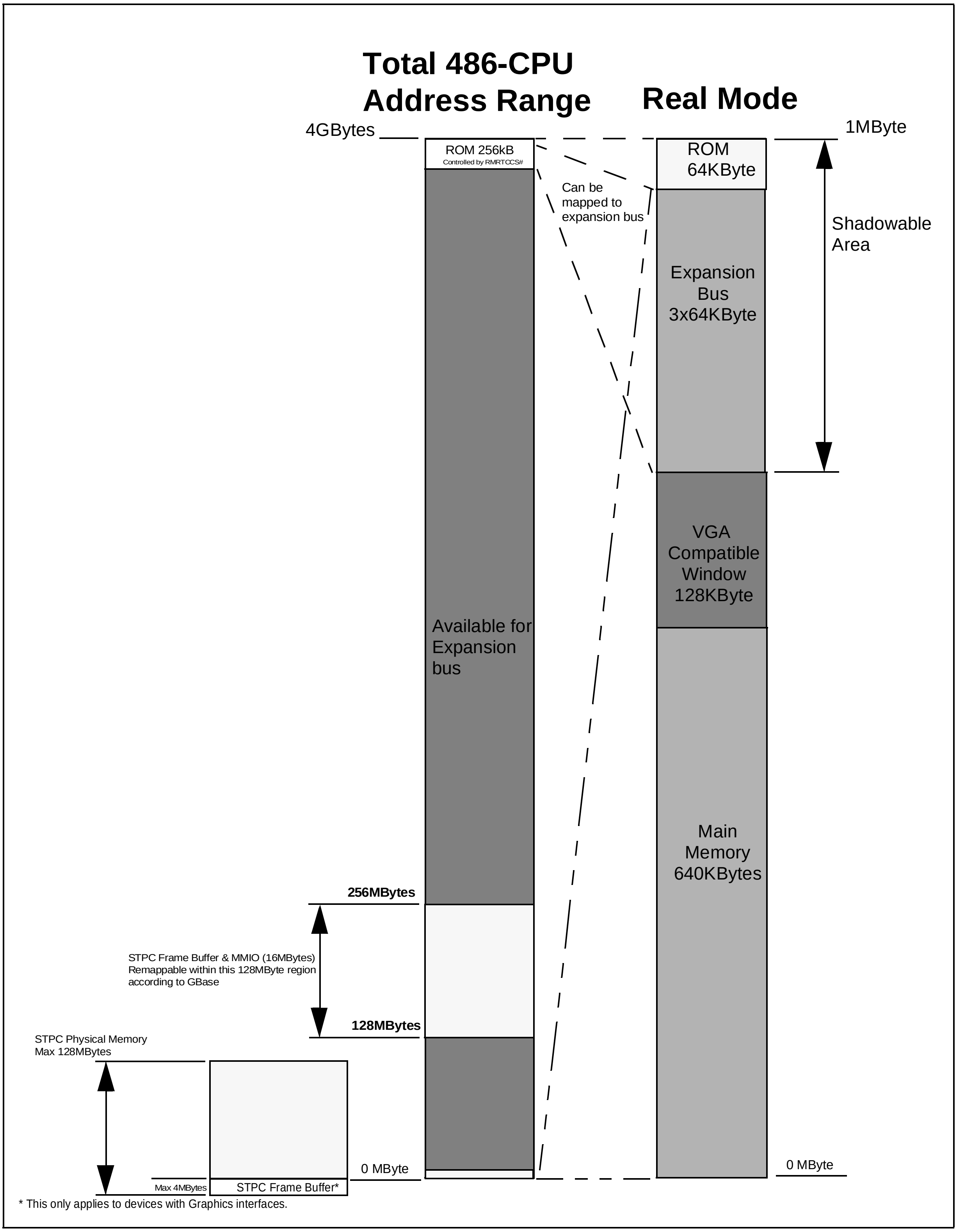


Figure 6-2. STPC Physical Memory Map

6.2. AGENT DECODING

All agents are decoded on a priority basis for instructions that are sent from the Host onto the host bus. If no agent on the Host Bus claims the cycle, it is then taken by the Host to PCI bridge (PCI North Bridge). If no agent on the PCI bridge claims the cycle it is forarded to the ISA bridge.

For PCI Memory accesses, the cycle is forwarded to the Host Bridge to be decoded in by the SDRAM Controller. PCI Master cycles follow the procedure above. For ISA Master devices, the cycle is first forwarded to the PCI Bridge and follows the procedure described above. ISA Memory cycles are forwarded in the same way as the PCI Memory cycles.

6.3. MEMORY ADDRESS MAP

Figure 6-2 illustrate the STPC Memory Map including the general overview of how the SDRAM controller is situated within the complete map including the STPC Frame Buffer Location.

Memory Region	Address Range	Description
MAIN MEMORY (640K)	00000000h 0009FFFFh	Host access maps to the main memory and no ISA or PCI cycle will be initiated. PCI master cycles in this range maps to main memory provided they are not claimed by a PCI Slave. The STPC relies on subtractive decode before initiating an internal memory cycle. ISA master cycles in this range maps to main memory. The STPC will negate IOCHRDY if necessary. The DMA master cycles in this range maps to main memory. The STPC will actively drive the SD bus during target reads and modify main memory for target write transfers. This address segment is considered always cacheable in the L1 cache. PCI and ISA master cycles in this range, require the L1 cache.
VGA FRAME BUFFER (128K)	000A0000h 000BFFFFh	This 128K address segment contains the VGA Frame buffer. Normally this address segment is mapped to the DOS frame buffer located in the main memory. However, if VGA is disabled or the VGA memory map mode is such that the VGA does not occupy the entire 128K address range, the host cycle is forwarded to the PCI bus and if not claimed by a PCI slave, it is further forwarded to the ISA bus. The PCI master cycles in this range, if not claimed by a PCI slave, will be mapped to the main memory or will be forwarded to the ISA bus as per the VGA decode described above. Similarly, the ISA or DMA master cycles will either map to the main memory or will be forwarded to the PCI. If no PCI slave claims the cycle, the STPC assumes existence of an ISA memory device at this address range. This segment is never cacheable.
SHADOW (16K)	000C0000h 000C3FFFh	This 16K address segment can be programmed software to either map to main memory or expansion buses. Further, reads and writes can have different mappings. If mapped to main memory, this segment will behave as the 0-640K segment. If not mapped to main memory, refer to Section 6.2. above If mapped to the main memory, the cacheability of this address range is controlled by software. If mapped to the ISA bus, the ROMCS# signal may optionally be asserted as controlled by software. This allows the system and video/peripheral BIOS to physically reside in a single ROM device.

Memory Region	Address Range	Description
SHADOW (16K)	000C4000h 000C7FFFh	This range has the same characteristics as that of 000C0000h-000C3FFFh segment, as described above. The shadow control for this address range is provided via Shadow Control register 0 and cacheability and ROM chip-select control via Shadow Control register 3.
SHADOW (16K)	000C8000h 000CBFFFh	This range has the same characteristics as that of 000C0000h-000C3FFFh segment as described above, with the exception of the cacheability attribute. This address range is hardwired to be non-cacheable. Shadow control for this address range is provided via Shadow Control register 0 and ROM chip-select control via Shadow Control register 3.
SHADOW (16K)	000CC000h 000CFFFFh	This range has the same characteristics as that of 000C8000h-000CBFFFh segment, as described above. Shadow control for this address range is provided via Shadow Control register 0 and ROM chip-select control via Shadow Control register 3. This address range is hardwired to be non-cacheable.
SHADOW (64K)	000D0000h 000DFFFFh	This range has the same characteristics as that of 000CC000h-000CFFFFh segment, as described above. Shadow control for this address range is provided via Shadow Control register 1 and can be controlled at 16K resolution. ROM chip-select generation for the entire 64K range can be controlled via Shadow Control register 3. This address range is hardwired to be non-cacheable.
SHADOW (64K)	000E0000h 000EFFFFh	This range has the same characteristics as that of 000CC000h-000CFFFFh segment ,as described above. Shadow control for this address range is provided via Shadow Control register 2 and can be controlled at 16K resolution. ROM chip-select generation for the entire 64K range can be controlled via Shadow Control register 3. This address range is hardwired to be non-cacheable.
SHADOW (64K)	000F0000h 000FFFFFh	This range has the same characteristics as that of 000C0000h-000C7FFFh segment, as described above. Shadow control for this address range is provided via Shadow Control register 3. If not shadowed in the main memory, cycles in this address range which are forwarded to the ISA bus will always results in an ROMCS# assertion. The cacheability of this address segment is controlled via Shadow Control register 3.
TOP OF ADDRESSABLE SDRAM MEMORY (1M)	00100000h	This address segment is mapped to the main memory with the exception of one hole that can optionally be opened in the range 1MBytes to 16 Mytes. The address range defined for the hole is mapped to the expansion buses and is described later in this section. The addressable SDRAM memory can be different from the populated memory due to the memory remapping and the frame buffer. This is described in more detail in a later section. With the exception of the memory holes, this address range has the same characteristics as the 0-640K (compatible DOS memory) range.
TOP OF ADDRESSABLE SDRAM MEMORY (4G-256K)	FFFC0000h	All cycles above the addressable SDRAM memory are forwarded to the expansion buses. Host access in this range initiates a PCI cycle and if unclaimed by a PCI slave, they are forwarded to ISA. Note that the ISA address space is only 16M. Higher addresses are aliased to this 16M space.
ROM ALIAS (4G-64K)	FFFF0000 FFFFFFFFh	This address segment is an alias of the 64K segment located at F0000h-FFFFFh and has the same attributes except that this segment can never be shadowed into the SDRAM memory. This is also true for address E0000h, D0000h and C0000h provided I/O register Index 51h (see Section 9.5.2.) is set correctly.

6.3.1. EXTENDED GRAPHICS SEGMENT

A 16M segment of memory anywhere between Top of addressable SDRAM memory and 256M can be optionally enabled via extended VGA Graphics Registers (GRA). This segment is located at 16M granularity. Refer to the Graphics section for a more detailed description of the layout of this memory segment.

Host access to this region is absorbed by the STPC and are either consumed internally, or initiate a frame buffer memory access.

PCI master access to this region, if not claimed by a PCI slave is absorbed by the STPC and treated the same way as a host access.

This address range by definition is not accessible to ISA and DMA masters, since it must be located at a 16M granularity above the addressable SDRAM memory. The ISA and DMA masters can access only up to 16M address range.

This address segment is always considered non-cacheable.

6.3.2. MEMORY HOLE

The Memory Hole register allows the creation of a hole in the memory space in 1-16M address range. This hole allows mapping expansion bus cards in the AT compatible address range when the addressable main memory size exceeds 16M. A host/PCI/ISA/DMA master cycle in this address range is handled in the same way as a cycle above the addressable memory range described above.

6.3.3. SMM MEMORY

The STPC uses the physical memory behind the CPU address range A0000h - B0000h for the SMM memory. The SMM base address register inside CPU needs to be programmed to A0000h. The initialization of the SMM memory is controlled by RAM System management register and redirects the CPU A0000h-B0000h address range to SMM memory. After the initialization, SMM memory can only be accessed when SMIACT# is active. The cacheability of this segment is hardwired to 0.

6.3.4. ADDRESSABLE SDRAM MEMORY

Addressable SDRAM memory is a function of the size of populated SDRAM, the size of graphic memory, the size of memory hole, and the shadow control of D0000h-DFFFFh and E0000h-EFFFFh segments.

TOPM = The size of total physical SDRAM is defined by SDRAM Bank 3 Register.

TOGM = The size of graphic memory is defined by Graphic memory size register.

MHOLE_SIZE = The size of memory hole defined by Memory Hole Control register.

REMAP_SIZE = 128KB, if none of the 8 x 16KB-segments of D0000h-EFFFFh is enabled for shadow, or 0KB, if any of the 8 x 16KB-segments of D0000h-EFFFFh is enabled for shadow.

The addressable SDRAM memory =

TOPM - TOGM + MHOLE_SIZE + REMAP_SIZE

6.3.5. CPU ADDRESS TO SDRAM ADDRESS MAPPING

The STPC implements a single memory subsystem for both the system as well as the frame buffer memory. In other words, the size of the SDRAM available to the system is reduced by the size of the SDRAM allocated to the frame buffer.

The CPU's concept of a physical address is a logical address to the STPC and is remapped to a SDRAM physical address. This section refers to the CPU's physical address as the "CPU address" and to the SDRAM's physical address as the "SDRAM address".

The lower range of the SDRAM, starting from the SDRAM address 00h, is allocated to frame buffer. The rest of the memory is used by the system. The CPU address is mapped to the SDRAM address space above the frame buffer address space. Since the size of the frame buffer can vary and is controlled by the Graphics Memory Size Register (Index 36 of the STPC configuration registers).

STPC also defines a memory hole to allow the existence of memory devices on the PCI or ISA buses. The size of the CPU address space is increased by these memory holes, if they exist. CPU address space D0000h to EFFFFh is mapped to the add-in card BIOS area. If this ROM space is not shadowed, then the CPU address space is increased by another 128 KBytes (also see Section 6.6.1.).

For example:

Total populated SDRAM = 4 MBytes

Frame buffer size = 256 KBytes*

Memory hole size = 1 MByte

Memory hole starting address = 200000h

Shadow feature for D0000h to EFFFFh = disabled

The total CPU memory = 4 MBytes - 256 KBytes + 1 MByte + 128 KBytes = 4 MBytes plus 896 KBytes

Since the frame buffer is 256 KBytes*, the system memory is reduced by 256 KBytes and becomes 3 MBytes plus 768 KBytes. Since a 1 MByte memory hole exists, the CPU address space is increased by 1 MByte and becomes 4 MBytes plus 768 KBytes. The CPU address between 3 MBytes plus 768 KBytes and 128 MBytes above this is mapped to the memory hole.

Since the shadowing of the CPU address range D0000h to EFFFFh reserved for add-on card BIOS is not enabled, the CPU memory is increased by 128 KBytes to make use of this SDRAM space that no device accesses. The total CPU memory then becomes 4 MBytes plus 896 KBytes.

* Not applicable to STPC Elite or Vega.

6.4. IO ADDRESS MAP

.

Table 6-1. IO Map Space

IO address	Description	Notes
0000h-000Fh	8237 DMA controller 1 registers.	1
0020h-0021h	8259 Interrupt controller 1 registers.	
0022h	STPC specific configuration registers index port	
0023h	STPC specific configuration registers data port	
0040h-0043h	8254 Timer/Counter registers.	1
0060h-0064h	Keyboard shadow registers.	1
0070h-0071h	NMI Mask control registers.	1
0080h-008Fh	DMA Page registers.	
0094h	Mother-board VGA enable.	2
00A0h-00A1h	8259 Interrupt controller 2 registers.	1
061h	ISA standard Port B.	1
00C0h-00DFh	8237 DMA controller 2 registers.	1
0102h	VGA setup register.	
03B4h,03B5h,03BAh	VGA registers.	
03D4h,03D5h,03DAh		
03C0h-03CFh		
0CF8h	PCI configuration Address register.	
0CFCh-0CFFh	PCI configuration Data register.	
46E8h	VGA add-in mode enable register.	2

The STPC implements a number of registers in IO address space. This is visible in the registers with access = 0022h/0023h. These regsters use the index and data programming system where the index to which the data is to be written to is programmed in register 0022h and the data is written to register 0023h. For an example on how this is implemeted, refer to Section 6.11.

These register occupy the map in the IO space in the table above:Table 6-1

Notes:

1. This address range is partially decoded. Refer to the Register Description section for more details.

2.This address is occupied only if the STPC is strapped to look like a mother-board VGA.

6.4.1. PCI CONFIGURATION ADDRESS MAP:

The STPC occupies Device number 0 slot on the PCI bus and implements a number of registers in PCI configuration address space. These registers occupy the following map (seeTable 6-2) :

Table 6-2. PCI Configuration Address Space

Offset	Description
00h-01h	Vendor Identification register
02h-03h	Device Identification register
04h-05h	PCI Command register
06h-07h	PCI Status register
08h	PCI Revision ID register
40h	PCI Control register

6.5. CACHE RELATED REGISTERS

The STPC supports two caching modes, write-through and write-back. For both modes, sdram read accesses are copied into the cache, and future accesses then return the cache copy with no access to sdram. The situation is different for write access; for the write-through mode, a write updates both the sdram and the cache, whereas in the the write-back mode, a write updates the cache only with the sdram update taking place later. The write-back mode offers improved performance over the write-through mode.

Two cache levels are generally available, level 1 (L1), within the CPU core, and level 2 (L2), within the chipset core (between the CPU and the sdram Controller). For the STPC product range, the L2 cache controller is included but as the required external connection pins are not provided, it is not usable. The L2 cache control registers described below are however used to configure the CPU L1 Cache architecture.

6.5.1. CACHE ARCHITECTURE REGISTER 0

This register controls various attributes of the L2 and L1 cache.

Cash_Arc0 Access = Regoffset = 0x20h

7	6	5	4	3	2	1	0
CPU PAS	BAO	L1 WB	SRAM		L2 B	L2 WBC	L2 BC
Default value after reset = 00000000h							

Bit Number	Mnemonic	Description
Bit 7	CPU PAS	**CPU pipelined access.** : 0: Not supported 1: Supported
Bit 6	BAO	**Burst addressing order.** 0: Intel 1: Linear
Bit 5	L1 WB	**L1 write back indication.** 0: Not supported 1: Supported
Bit 4-3	SRAM	**SRAM type.** These bits control the type of SRAMs used to construct L2 cache. (See Table 6-3)
Bit 2	L2 B	**Number of L2 banks.** When programmed to 2 banks, L2 interleaving is enabled. 0: One bank 1: Two banks
Bit 1	L2 WBC	**L2 write back control.** 0: Write through 1: Write back
Bit 0	L2 BC	**L2 cache enable.** 0: Disabled 1: Enabled

Table 6-3. Bits 4-3 SRAM Type

Bit 4	Bit 3	L2 cache SRAM type
0	0	asynchronous SRAM
0	1	synchronous burst SRAM
1	0	synchronous burst pipelined SRAM
1	1	reserved

6.5.2. CACHE ARCHITECTURE REGISTER 1

This register controls various attributes of L2 cache.

Cash_Arc1 Access = Regoffset = 0x21h

7	6	5	4	3	2	1	0
L2 CS			IO NA	S FIFO		R AWE	Rsv
Default value after reset = 00000000h							

Bit Number	Mnemonic	Description
Bits 7-5	L2 CS	**L2 cache size.** (See Table 6-4)
Bit 4	IO NA#Enable	**IO NA# Enable.** 0: Generate NA# 1: Don't generate NA#
Bits 3-2	S FIFO	**Source FIFO low water mark.** These bits control the degree of concurrency between a L1 cache line fill and start of the next memory access. A cache line wide read buffer is implemented. Due to pipelining, it is possible that the buffer may be filled up ahead of drain. Then if the next access is also a read from memory, these bits determine when the next read will be kicked off relative to the drain of the current line from the read buffer. The optimal value is a function of the drain rate of the buffer which depends on the cache RAM type and the programmed burst parameters. A value of ‘0’ for this field is the least optimal value but will always work. (See Table 6-5)
Bit 1	R AWE	**Read around write enable.** 0: Reads can not proceed around any posted writes 1: Reads can go around a posted write if it is to a different address to the posted writes
Bit 0	Rsv	**Reserved.**

Table 6-4. L2 Cache Size

Bit 7	Bit 6	Bit 5	L2 Cache Size
0	0	0	64Kb
0	0	1	128Kb
0	1	0	256Kb
0	1	1	512Kb
1	0	0	1 MB
1	0	1	2 MB

.

Table 6-5. Source FIFO Low Water Mark

Bit 3	Bit 2	Start next read...
0	0	only after completely finishing current fill
0	1	when 1 QWORD is still to be emptied
1	0	when 2 QWORDs are still to be emptied
1	1	when 3 QWORDs are still to be emptied

6.5.3. CACHE ARCHITECTURE REGISTER 2

Cash_Arc2 Access = Regoffset = 0x22h

7	6	5	4	3	2	1	0
Rsv	SHDD	CWEPW	CDHAWE	BAWS		TAWS	
Default value after reset = 11111111h							

Bit Number	Mnemonic	Description
Bit 7	Rsv	**Reserved.**
Bit 6	SHDD	**Slow host data driver.** 0: Fast, One clock to drive the HD bus 1: Slow, two clocks to drive HD bus
Bit 5	CWEPW	**Cache write enable pulse width.** 0: 1 clock wide 1: 1.5 clocks wide Applicable to asynchronous SRAMs only. Must be ‘0’ for synchronous SRAMs.
Bit 4	CDHAWE	**Cache data hold after write enable.** 0: Data removed in the same clock as write enable trailing edge 1: Data is kept valid for 1 extra clock after write enable Must be a ‘1’ if 1.5 clocks wide write enable pulse width is selected via bit 5 above.
Bits 3-2	BAWS	**Burst access wait states.** (See Table 6-6)
Bits 1-0	TAWS	**Tag access wait states.** (See Table 6-7)

Table 6-6. Burst Access Wait States

Bit 3	Bit 2	Burst access wait states
0	0	fastest
0	1	1 clock slower than fastest
1	0	2 clocks slower than fastest
1	1	3 clocks slower than fastest

Table 6-7. Tag Access Wait States

Bit 1	Bit 0	Tag access wait states
0	0	fastest
0	1	1 clock slower than fastest
1	0	2 clocks slower than fastest
1	1	3 clocks slower than fastest

6.6. ADDRESS DECODE RELATED REGISTERS

The following registers are all 8-bit. They are accessed by setting the Configuration Index Port (22h) to the Configuration Index (C.I.) shown, and then reading or writing the appropriate values from the Configuration Register Data Port (23h).

6.6.1. MEMORY HOLE CONTROL REGISTER

This 8-bit register defines the enable, size, and starting address of memory hole. Any memory accesses to this memory hole are directed to PCI/ISA bus.

MEM_HOLE Access = Regoffset = 0x24h

7	6	5	4	3	2	1	0
MHE	MHS			MHSA			
Default value after reset = 00000000h							

Bit Number	Mnemonic	Description
Bit 7	MHE	**Memory Hole Enable.** This bit controls the enable of memory hole function. 0 = disabled 1 = enabled
Bits 6-4	MHS	**Memory Hole Size.** These bits control the size of memory hole (See Table 6-8)
Bits 3-0	MHSA	**Memory Hole Start Address.** These bits control the bits 23-20 of the memory hole starting address. The memory hole starting address must be aligned to the hole size.

Table 6-8. Memory Hole Size

Bit 6	Bit 5	Bit 4	Memory Hole Size
0	0	0	1 MB
0	0	1	2 MB
0	1	1	4 MB
1	1	1	8 MB
others			reserved

Programming notes:

This memory hole is also non-cacheable.

6.6.2. SHADOW CONTROL REGISTER 0

This 8-bit register controls the read/write attributes of the memory located at C0000h-CFFFFh. Each 16k of the whole 64k is controlled by 2 bits, one for read and one for write.

SHADOW_0 Access = Regoffset = 0x25h

7	6	5	4	3	2	1	0
RC1	WC1	RC2	WC2	RC3	WC3	RC4	WC4
Default value after reset = 00000000h							

Bit Number	Mnemonic	Description
Bit 7	RC1	**Read Control CC000h-CFFFFh.** This bit controls the read attribute of the CC000h-CFFFFh memory. 0 = shadow disabled for read cycle 1 = shadow enabled for read cycle
Bit 6	WC1	**Write Control CC000h-CFFFFh.** This bit controls the write attribute of the CC000h-CFFFFh memory. 0 = shadow disabled for write cycle 1 = shadow enabled for write cycle
Bit 5	RC2	**Read Control C8000h-CBFFFh.** This bit controls the read attribute of the C8000h-CBFFFh memory. 0 = shadow disabled for read cycle 1 = shadow enabled for read cycle
Bit 4	WC2	**Write Control C8000h-CBFFFFh.** This bit controls the write attribute of the C8000h-CBFFFh memory. 0 = shadow disabled for write cycle 1 = shadow enabled for write cycle
Bit 3	RC3	**Read Control C4000h-C7FFFh.** This bit controls the read attribute of the C4000h-C7FFFh memory. 0 = shadow disabled for read cycle 1 = shadow enabled for read cycle
Bit 2	WC3	**Write Control C4000h-C7FFFFh.** This bit controls the write attribute of the C4000h-C7FFFh memory. 0 = shadow disabled for write cycle 1 = shadow enabled for write cycle

Bit Number	Mnemonic	Description
Bit 1	RC4	**Read Control C0000h-C3FFFh.** This bit controls the read attribute of the C0000h-C3FFFh memory. 0 = shadow disabled for read cycle 1 = shadow enabled for read cycle
Bit 0	WC4	**Bit 0 Write Control C0000h-C3FFFFh.** This bit controls the write attribute of the C0000h-C3FFFFh memory. 0 = shadow disabled for write cycle 1 = shadow enabled for write cycle

Programming Notes:

There is single cacheability bit for the 32k Video BIOS segment (C0000h-C7FFFh) located in Shadow Control register 2. C7FFFFh-CFFFFh segment has the cacheability bit hardwired to ‘1’ (enabled). If shadow is enabled for read/write cycles, read from and write to this area are directed to the system memory. Or else the cycles are forwarded to the expansion buses.

6.6.3. SHADOW CONTROL REGISTER 1

Similar to Shadow Control Register 0, this 8-bit register controls the read/write attributes of the memory located at D0000h-DFFFFh.

SHADOW_1 Access = Regoffset = 0x26h

7	6	5	4	3	2	1	0
SRC	SWC	SWC	SWC	SRC	SWC	SRC	SWC
Default value after reset = 00000000h							

Bit Number	Mnemonic	Description
Bit 7	SRC	**Shadow Read Control DC000h-DFFFFh.** This bit controls the read attribute of the DC000h-DFFFFh memory. 0 = shadow disabled for read cycle 1 = shadow enabled for read cycle
Bit 6	SWC	**Shadow Write Control DC000h-DFFFFh.** This bit controls the write attribute of the DC000h-DFFFFh memory. 0 = shadow disabled for write cycle 1 = shadow enabled for write cycle
Bit 5	SWC	**Shadow Write Control D8000h-DBFFFh.** This bit controls the read attribute of the D8000h-DBFFFh memory. 0 = shadow disabled for read cycle 1 = shadow enabled for read cycle
Bit 4	SWC	**Shadow Write Control D8000h-DBFFFh.** This bit controls the write attribute of the D8000h-DBFFFh memory. 0 = shadow disabled for write cycle 1 = shadow enabled for write cycle
Bit 3	SRC	**Shadow Read Control D4000h-D7FFFh.** This bit controls the read attribute of the D4000h-D7FFFh memory 0 = shadow disabled for read cycle 1 = shadow enabled for read cycle
Bit 2	SWC	**Shadow Write Control D4000h-D7FFFh.** This bit controls the write attribute of the D4000h-D7FFFh memory. 0 = shadow disabled for write cycle 1 = shadow enabled for write cycle
Bit 1	SRC	**Shadow Read Control D0000h-D3FFFh.** This bit controls the read attribute of the D0000h-D3FFFh memory. 0 = shadow disabled for read cycle 1 = shadow enabled for read cycle
Bit 0	SWC	**Shadow Write Control D0000h-DFFFFh.** This bit controls the write attribute of the D0000h-DFFFFh memory. 0 = shadow disabled for write cycle 1 = shadow enabled for write cycle

Programming Notes: This entire 64K segment has the cacheability bit hardwired to ‘0’ (disabled)

6.6.4. SHADOW CONTROL REGISTER 2

Similar to Shadow Control Register 0, this 8-bit register controls the read/write attributes of the memory located at E0000h-EFFFFh.

SHADOW_2 Access = Regoffset = 0x27h

7	6	5	4	3	2	1	0
RC	WC	RC	WC	RC	WC	RC	WC
Default value after reset = 00000000h							

Bit Number	Mnemonic	Description
Bit 7	RC	**Read Control EC000h-EFFFFh.** This bit controls the read attribute of the EC000h-EFFFFh memory. 0 = shadow disabled for read cycle 1 = shadow enabled for read cycle
Bit 6	WC	**Write Control EC000h-EFFFFh.** This bit controls the write attribute of the EC000h-EFFFFh memory. 0 = shadow disabled for write cycle 1 = shadow enabled for write cycle
Bit 5	RC	**Read Control E8000h-EBFFFh.** This bit controls the read attribute of the E8000h-EBFFFh memory. 0 = shadow disabled for read cycle 1 = shadow enabled for read cycle
Bit 4	WC	**Write Control E8000h-EBFFFFh.** This bit controls the write attribute of the E8000h-EBFFFh memory. 0 = shadow disabled for write cycle 1 = shadow enabled for write cycle
Bit 3	RC	**Read Control E4000h-E7FFFh.** This bit controls the read attribute of the E4000h-E7FFFh memory. 0 = shadow disabled for read cycle 1 = shadow enabled for read cycle
Bit 2	WC	**Write Control E4000h-E7FFFFh.** This bit controls the write attribute of the E4000h-E7FFFh memory. 0 = shadow disabled for write cycle 1 = shadow enabled for write cycle
Bit 1	RC	**Read Control E0000h-E3FFFh.** This bit controls the read attribute of the E0000h-E3FFFh memory. 0 = shadow disabled for read cycle 1 = shadow enabled for read cycle
Bit 0	WC	**Bit 0 Write Control E0000h-EFFFFh.** This bit controls the write attribute of the E0000h-EFFFFh memory. 0 = shadow disabled for write cycle 1 = shadow enabled for write cycle

Programming Notes: This entire 64K segment has the cacheability bit hardwired to ‘0’ (disabled)

6.6.5. SHADOW CONTROL REGISTER 3

This 8-bit register controls the cacheability attributes of C0000h-C7FFFh and F0000h-FFFFFh shadow segments.

SHADOW_3 Access = Regoffset = 0x28h

7	6	5	4	3	2	1	0
SMRAM	CCF	CCC	Rsv			RCF	WCF
Default value after reset = 00000000h							

Bit Number	Mnemonic	Description
Bit 7	SMRAM	**SDRAM Initialization Enable.** This bit controls whether CPU accesses in A0000h-BFFFFh address range are decoded as VGA frame buffer access or SMRAM access. 0 = A0000h-BFFFFh is interpreted as VGA frame buffer access 1= A0000h-BFFFFh is interpreted as SMRAM access. The STPC allows for 128KBytes of SMRAM. Physically this memory is located in the system memory behind the higher address range. This area of the system memory is normally unused since this address range is normally mapped to frame buffer which has its own memory. When the CPU is operating in SMM, accesses in the range of A0000-BFFFFh goes to SMRAM instead of VGA frame buffer. The rest of the address map remains unchanged. The address range A0000h-BFFFFh is always non-cacheable.
Bit 6	CCF	**Cache Control F0000h-FFFFFh.** This bit controls the cacheability of F0000h-FFFFFh block when the shadow function is enabled. 0 = cacheability disabled 1 = cacheability enabled
Bit 5	CCC	**Cache Control C0000h-C7FFFh.** This bit controls the cacheability of C0000h-C7FFFh block when the shadow function is enabled. 0 = cacheability disabled 1 = cacheability enabled
Bits 4-2	Rsv	**Reserved.**
Bit 1	RCF	**Read Control F0000h-FFFFFh.** This bit controls the read attribute of F0000h-FFFFFh memory. 0 = shadow disabled for read cycle 1 = shadow enabled for read cycle
Bit 0	WCF	**Bit 0 Write Control F0000h-FFFFFh.** This bit controls the write attribute of F0000h-FFFFFh memory. 0 = shadow disabled for write cycle 1 = shadow enabled for write cycle

Programming notes:

The rest of the shadow RAM segments have the cacheability bits hardwired to ‘0’ (disabled). This register also provides control over the address range for which ROM chip-select (ROMCS#) will be asserted allowing various BIOSes (system, video, disk etc.) to be implemented in a single part. Bit 7 of this register also provides accessibility to the SMM mode RAM (SMRAM).

6.6.6. VGA DECODE REGISTER

This 8-bit register controls address decode for the internal VGA as follows:

VGA_DEC Access = Regoffset = 0x29h

7	6	5	4	3	2	1	0
Rsv		Rsv		DC	PSE	I VGA D	ADE
Default value after reset = 00000011h							

Bit Number	**Mnemonic**	**Description**
Bits 7-6	Rsv	Reserved.
Bit 5-4	Rsv	Reserved.
Bit 3	DC	stop d-clock (Dot clock).
Bit 2	PSE	Palette Snoop Enable: 1 =Palette write cycles are propagated to PCI bus in addition to updating the internal palette. 0 = Palette write cycles are terminated internally and are not propagated to PCI.
Bit 1	I VGA D	**Internal VGA Disable.** This bit if set to a ‘0’ will disable internal VGA. Otherwise if set to a ‘1’, it will enable the internal VGA.
Bit 0	ADE	**Add-in Decode Enable.** This bit if set to a ‘0’ will map the internal VGA to add-in card address space. Otherwise if set to a ‘1’ it will map the VGA to mother-board address space.

6.7. HOST SDRAM CONTROLLER REGISTERS

The STPC manages 4 Memory Banks (if DIMM sockets are used they can be populated with either single or double sided 64-bit data DIMMs). For SDRAM densities are supported see the datasheet Section 6.3.3.
.

Configuration registers 30-33 provide the top addresses for each bank. Any bank can be skipped by the top addresses of two consecutive banks having the same address.

6.7.1. MEMORY BANK 0 REGISTER - C.I. 30H (MEMORY__BANK0)

This 8-bit register controls the top address of memory bank 0. Register bit 7-0 corresponding to memory address bits 27-20.

Bank 0 Top Address = Memory Bank0 size in MBytes -1.

Bank 1 Top Address = Memory Bank0 + Memory Bank1 size in MBytes -1

This register defaults to 07h.

Example 1:

Memory Bank0 = 4MB
Memory Bank1 = 4MB

Bank 0 Top Address = 4 -1= 3= 03h

Bank 1 Top Address = 4 + 4 - 1 = 07h

Bank 2, 3 Top Address = 07h

Example 2: for use with double sided DIMMs

Memory Bank0 = 32MBytes (dbl. sided DIMMS)
Memory Bank1 = 32MBytes (dbl. sided DIMMS)

Bank 0 Top Address = 16 - 1 = 15 = 0Fh

Bank 1 Top Address = 16 + 16 - 1 = 31 = 1Fh

Bank 2 Top Address = 32 + 16 - 1 = 47 = 2Fh

Bank 3 Top address = 48 + 16 - 1 = 63 = 3Fh

6.7.2. MEORY BANK 1 REGISTER - C.I. 31H (MEMORY_BANK1)

This register controls the top address of memory bank 1.

6.7.3. MEMORY BANK 2 REGISTER - C.I. 32H (MEMORY_BANK2)

This register controls the top address of memory bank 2.

6.7.4. MEMORY BANK 3 REGISTER - C.I. 33H (MEMORY_BANK3)

This register controls the top address of memory bank 3.

6.7.5. GRAPHICS MEMORY SIZE REGISTER

This register defines the size of SDRAM used by the graphics for frame buffer.

GRAPH_MEM Access = Regoffset = 0x36h

7	6	5	4	3	2	1	0
GRAS	Rsv	TGM					
Default value after reset = 00000100h							

Bit Number	Mnemonic	Description
Bit 7	Rsv	**Reserved**
Bit 6	Rsv	Reserved.
Bits 5-0	TGM	Top of Graphics Memory. This indicates frame buffer size in 128KB units. The range is 0 to 32 for 0 to 4MB framebuffer, so 6 bits are necessary.

6.7.6. SDRAM REFRESH REGISTER

This refresh register also contains a number of host clock settings for the SDRAM refresh interval.

SDRAM_Ref Access = 0022h/0023h Regoffset = 039h

7	6	5	4	3	2	1	0
RE	RC						
Default value after reset = 30h							

Bit Number	Mnemonic	Description
Bit 7	RE	**Refresh Enable.** This bit must be programmed to ‘0’ for normal operation
Bits 6-0	RC	Refresh Cycle. (HCLK frequency in MHz * 15.6us) >> 4

* Examples: (rounded down to nearest integer)

round_down((75MHz * 15.6us) >> 4) = 73 = 49h

round_down((66MHz * 15.6us) >> 4) = 65 = 41h

round_down((60MHz * 15.6us) >> 4) = 58 = 3Ah

round_down((50MHz * 15.6us) >> 4) = 48 = 30h

Programming notes:

The refresh interval should be reset to the smallest likely run time value (typically 48 HCLKs) to provide warm up cycles for the SDRAM.

A refresh request is generated whenever this register is written to without setting the refresh enable bit.

6.7.7. PRESENTS DETECT REGISTER - C.I. 97H

This register is read through the DDC register in Section 11.9.30. .

6.8. WATCHDOG TIMER

6.8.1. INTRODUCTION

In real time systems, many software programs need a feature to prevent a process going into an endless loop, leading to a possible system failure. One way to achieve this is to use a watchdog timer. When a possible endless loop is detected, the watchdog timer is set and starts counting. If the possible endless loop does not terminate by itself, the resulting watchdog counter timeout is used to bring control back to the watchdog interrupt handler, which can then run a suitable program to terminate the process.

A watchdog timer is thus used to protect against system failures by providing a means of escape from unexpected input conditions, external events or programming errors. Once the watchdog timer has been started, it must be cleared by software on a regular basis so that under normal conditions it never reaches its timeout value. If however, it does reach the timeout value, it may be assumed that a system failure has occurred, and steps can then be taken to recover the system.

When the watchdog counter reaches the timeout value, an interrupt (NMI, SMI or INTR) will be generated and the CPU will start executing an Interrupt Service Routine (ISR). Here the program will read the status register of the watchdog timer to check the interrupt status bits. The routine will now run the programmed instructions to recover the system.

Note: A status register read cycle will clear the Interrupt (NMI, SMI or INTR) bits.

6.8.2. WATCHDOG TIMER FEATURES

- 16-bit Programmable Clock Divider.
- 32-bit Programmable Down Counter.
- Interrupt Generation.
- Start/Stop control of Watchdog operation.

6.8.3. WATCHDOG TIMER REGISTERS

Table 6-9. Watchdog Timer Registers

Offset	Register Width	Register Name	Register Description	Default Value
0x00h	32-bit	Watchdog Clock Divider Register	To load clock divider value	0x00000000h
0x04h	32-bit	Watchdog Count Load Register	To load watchdog count value	0x00000000h
0x08h	32-bit	Watchdog Count Read Register	To read watchdog output	0x00000000h
0x0Ch	32-bit	Watchdog Control Register	To configure watchdog	0x00000000h
0x10h	32-bit	Watchdog Counter Enable Register	To enable watchdog	0x00000000h
0x14h	32-bit	Watchdog Status Register	To read watchdog status	0x00000000h

6.8.4. WATCHDOG CLOCK DIVIDER REGISTER

This 32-bit read/write register contains the clock divider value.

WDCLKDVDRG Access =Section 6.11. Regoffset = 00h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Clock divider value															
Default value after reset = 00000000h															

Bit Number	**Mnemonic**	Description
Bits 31-16	Rsv	Reserved
Bits 15-0	wdClk	The watchdog timer is clocked by Host Clock. The clock divider logic generates the wdClk signal with a maximum clock divide value of 2^{16} = 65536, minimum clock divide value of 1. As an example, if Host Clock frequency is 100 MHz and the Clock Divider Value is set to 0004h, then 25 MHz wdClk pulses will be generated. Formula: wdClk = Host Clock / Clock Divider Value **Note:** wdClk will be used as the ‘Enable’ signal for the watchdog counter. The clk for the counter will be Host Clock.

6.8.5. WATCHDOG COUNT LOAD REGISTER

This is a 32-bit read/write register into which the count value is loaded.

WDCNTLDRG Access =Section 6.11. Regoffset = 04h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Count value															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Count value															
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	cnt	Watchdog counter count value. It is required to be programmed by the user. A32-bit write cycle is used to load the count value. Maximum Count value = 2^{32} = 4Gbyte, Minimum count value = 1. Formula: WatchDog Time = Count Value * Divider value/ Host Clock frequency. The maximum and minimum watchdog counter times depend on the host clock, and can vary from Nano seconds to a number of days.

6.8.6. WATCHDOG COUNT READ REGISTER

This is a 32-bit read-only register into which the count value can be read.

WDCNTRDRG Access =Section 6.11. Regoffset = 08h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Current watchdog Counter Output Value															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Current watchdog Counter Output Value															
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	cntval	This register is used to read the counter value on the fly if required by the software.

6.8.7. WATCHDOG CONTROL REGISTER

This 32-bit read/write register is used to control the watchdog timer.

WDCTRLREG Access = Section 6.11. Regoffset = 0Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv														Interrupt	
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-2	Rsv	Reserved.
Bits 1-0	Interrupt	These two bits are used to route the watchdog interrupt signal to SMI, to NMI or to INTR, as follows:. 0 0: No output 0 1: SMI 1 0: NMI 1 1: INTR

6.8.8. WATCHDOG COUNTER ENABLE REGISTER

This 32-bit read/write register is used to enable the watchdog timer.

WDCOUNTERENABLEREG Access = Section 6.11. Regoffset = 10h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv															En
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-1	Rsv	Reserved.
Bit 0	En	Watchdog Counter Enable: This bit controls the watchdog counter operation. To enable and load the watchdog counter, the programmer must write a logic ‘1’ into this register. To disable the counting operation, the programmer must write a logic ‘0’ into this register. This register bit automatically goes from logic ‘1’ to logic ‘0’ when the watchdog counter expires and stops. To avoid watchdog timer expiration, the software should periodically write a ‘1’ into this register during the time the watchdog counter is enabled.

6.8.9. WATCHDOG STATUS REGISTER

This is the 32-bit read-only watchdog status register.

WDSTATUSREG Access = Section 6.11. Regoffset = 14h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv												INTR	NMI	SMI	TOut
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31- 4	Rsv	Reserved.
Bit 3	INTR	INTR status: A ‘1’ indicates that a watchdog interrupt has been generated and is routed to the INTR.
Bit 2	NMI	NMI status: A ‘1’ indicates that a watchdog interrupt has been generated and is routed to the NMI.
Bit 1	SMI	SMI status: A ‘1’ indicates that a watchdog interrupt has been generated and is routed to the SMI.
Bit 0	TOut	Watchdog Timeout status: This bit indicates whether or not the watchdog counter has expired. 1 denotes expired; 0 denotes not expired.

Note: Any read of this register will clear all status bits on completion of the read operation.

6.9. I/O ADDRESS TRAPPING

6.9.1. INTRODUCTION

For software debugging and I/O port emulation purposes, it is necessary for the system software to be able to trap an I/O access. Programming allows a trapped I/O cycle to be either terminated by the I/O Trapper or passed to the actual I/O port. Four 16-bit registers are provided to store the addresses to be trapped.

6.9.2. I/O TRAPPING REGISTERS

Table 6-10. I/O Trapping Registers

Offset	Register Width	Register Name	Register Description	Default Value
0x10h	32-bit	I/O Address Register *AdrReg0*	To load trap address	0x00000000h
0x14h	32-bit	I/O Address Register *AdrReg1*	To load trap address	0x00000000h
0x18h	32-bit	I/O Address Register *AdrReg2*	To load trap address	0x00000000h
0x1Ch	32-bit	I/O Address Register *AdrReg3*	To load trap address	0x00000000h
0x20h	32-bit	I/O Address Mask Register *MaskReg0*	To mask trap address bits	0x00000000h
0x24h	32-bit	I/O Address Mask Register *MaskReg1*	To mask trap address bits	0x00000000h
0x28h	32-bit	I/O Address Mask Register *MaskReg2*	To mask trap address bits	0x00000000h
0x2Ch	32-bit	I/O Address Mask Register *MaskReg3*	To mask trap address bits	0x00000000h
0x30h	32-bit	Trapped Address Register *TAReg*	To store trapped I/O address	0x00000000h
0x34h	32-bit	Trapped Data Register *TDReg*	To store trapped data value from data bus	0x00000000h
0x38h	32-bit	Trapped Byte Enable Register *TBEnReg*	To store trapped byte enable	0x00000000h
0x3Ch	32-bit	Control Register *CReg*	To control I/O trapper operation	0x00000000h
0x40h	32-bit	I/O Trap Wait-for-Ready Generation Register *IOWaitReg*	To load delay count value for Ready generation after I/O trap interrupt	0x00000000h
0x44h	32-bit	Status Register *SReg*	To read I/O trapper status	0x00000000h

6.9.3. I/O ADDRESS REGISTERS

The four read/write registers, *AdrReg0, AdrReg1, AdrReg2 and AdrReg3*, are programmed to select 16 bit I/O addresses, in byte format, to be trapped.

ADRREG# Access = 0022h/0023h Regoffset = see Table 6-10

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I/O Address to be trapped															
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-16	Rsv	Reserved.
Bits 15-0	AdrReg#	These registers are initialised before the application is run. Use the corresponding Mask Register to extend the address scope. If less than four addresses are to be trapped, the unused address register(s), together with the associated mask register(s), must be programmed with the same values as set in *AdrReg0* and *MaskReg0* respectively. This must be done before the I/O trapping facility is enabled.

6.9.4. IO MASK REGISTERS

The four read/write registers, *MaskReg0, MaskReg1, MaskReg2 and MaskReg3*, each provide eight bits to correspond with the least significant address bits in the associated Address Registers.

MASKREG# Access = 0022h/0023h Regoffset = see Table 6-10

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv								Mask Value for I/O address to be trapped							
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-8	Rsv	Reserved.
Bits 7-0	MaskReg#	Mask Value for I/O address to be trapped. Setting a mask register bit to 1 sets the corresponding address register bit to the “don’t care” state during the I/O address comparison operation.

6.9.5. TRAPPED ADDRESS REGISTER

This 32-bit read-only register is used to store a trapped address when an Address Trapped HIT occurs after address comparison. The register is read to determine a trapped address.

TADDREG Access = 0022h/0023h Regoffset = see Table 6-10

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Trapped Address															
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-16	Rsv	Reserved.
Bits 15-0	TAddReg	Trapped Address

6.9.6. TRAPPED DATA REGISTER

This 32-bit read-only register is used to store data from the Data Bus resulting from an address Trap HIT following an address comparison.

TDATREG# Access = 0022h/0023h Regoffset = see Table 6-10

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Trapped Data Value															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Trapped Data Value															
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	TDatReg	Trapped Data Value.

6.9.7. TRAPPED BYTE ENABLE REGISTER

This 32-bit read/write register is used to store the value on the Byte Enable bus resulting from an Address HIT following an address comparison.

TBEREG Access = 0022h/0023h Regoffset = see Table 6-10

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv												TBE			
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-4	Rsv	Reserved.
Bits 3-2	IRQRoute	These bits are used to control the I/O trap interrupt routing to SMI, NMI or INTR: Bit 3 Bit 2 0 0 No output 0 1 SMI 1 0 NMI 1 1 INTR
Bit 1	I/ODest	I/O Destination: 0: I/O cycle is passed to actual I/O port irrespective of trapping condition. 1: I/O cycle is terminated.
Bit 0	I/OTE	I/O Trap Enable: When at ‘1’, I/O trapping facility is enabled. Once an I/O cycle is trapped, this bit returns automatically to ‘0’. The disabled ‘0’ condition can be programmed.

6.9.8. CONTROL REGISTER

This 32-bit read/write register is used for I/O Trapping control purposes.

CREG Access = 0022h/0023h Regoffset = see Table 6-10

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv												IRQ Routing		CTL	TE
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-4	Rsv	Reserved.
Bits 3:2	IRQRoute	These bits are used to control the I/O trap interrupt routing to SMI, NMI or INTR as follows: Bit 3 **Bit 2** 0 0 No output 0 1 SMI 1 0 NMI 1 1 INTR
Bit 1		When at ‘1’ I/O cycle is terminated by the I/O Trapper and the cycle does not pass to the output. When at ‘0’ I/O cycle passes to I/O output port irrespective of trapping condition.
Bit 0		Trap Enable: When at ‘1’ I/O trapping is enabled. When an I/O cycle has been trapped, this bit returns automatically to ‘0’, otherwise it can be disabled by the programmer.

6.9.9. TRAP WAIT-FOR-READY GENERATION REGISTER

This 32-bit read/write register stores the delay count value for RDY generation after an I/O trap interrupt generation.

TWAITREG Access = 0022h/0023h Regoffset = see Table 6-10

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv								Delay Count Value							
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-8	Rsv	Reserved.
Bits 7-0	TWait	Delay count value.

6.9.10. STATUS REGISTER

This is the 32-bit read-only Status register. This register is cleared by any I/O read cycle.

STATREG# Access = 0022h/0023h Regoffset = see Table 6-10

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv											Status bits				
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-5	Rsv	Reserved.
Bit 4	INTR_Stat	'1' denotes that an I/O Trapper interrupt has been generated for application to the INTR pin.
Bit 3	NMI_Stat	'1' denotes that an I/O Trapper interrupt has been generated for application to the NMI pin.
Bit 2	SMI_Stat	'1' denotes that an I/O Trapper interrupt has been generated for application to the SMI pin.
Bit 1	R/W_Access	Trapped Read/Write access: This bit is only valid when bit [0] is at '1', i.e. when a cycle has been trapped. If the trapped cycle is I/O write, this bit goes to a '1'; If the trapped cycle is I/O read, this bit goes to '0'.
Bit 0	I/O_Stat	I/O Trap status: When an I/O cycle has been trapped, this bit goes to '1', otherwise it remains at '0'.

6.10. REAL-TIME 32-BIT MEMORY ADDRESSING

6.10.1. INTRODUCTION

Although 486 and Pentium processor devices have a 32-bit wide address bus, indicating a memory address range up to 4 GByte, in Real Mode, only the lowest 1 MByte is accessible and memory above 1 MByte can only be accessed in Protected Mode. The facility described in this section allows access to more than 1 MByte of memory in Real Mode, a useful feature for accessing memory mapped registers which reside above the 1 MByte range.

Note: This feature will only work when tacc is greater than 1.

6.10.2. REAL MODE CONFIGURATION REGISTERS

Table 6-11. Real Mode Configuration Registers

Offset	Register Width	Register Name	Register Description	Default Value
0x20h	32-bit	Real Mode Aperture Register *RMAReg*	To load Aperture address bits	0x00000000h
0x21h	32-bit	Remap Address Register *RAdrReg*	To store Remap address	0x00000000h
0x22h	32-bit	Control Register *CtlReg*	To control 32-bit Real Mode access operation	0x00000000h

6.10.3. REAL MODE APERTURE REGISTER

This 32-bit read/write register stores the Aperture address bits.

RMAREG Access = 0022h/0023h Regoffset = see Table 6-11

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv												Ap Address			
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Ap Address				Rsv											
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31- 20	Rsv	Reserved.
Bits 19 - 12	RMA	Aperture address bits for comparison when memory cycle is running in Real Mode with 32-bit real mode access enabled.
Bits 11 - 0	Rsv	Reserved

6.10.4. REMAP ADDRESS REGISTER

This 32-bit read/write register stores an address to transfer to the Real Mode Aperture Register when an Aperture HIT has been generated following an Aperture Address comparison.

RADREG Access = 0022h/0023h Regoffset = see Table 6-11

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Remap Address bits															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Remap Address bits				Rsv											
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31- 12	RMAd	Remap address bits
Bits 11 - 0	Rsv	Reserved

6.10.5. CONTROL REGISTER

This 32-bit read/write register is used to control 32-bit Real mode access.

CTLREG Access = 0022h/0023h Regoffset = see Table 6-11

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv															ARE
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31- 1	Rsv	Reserved.
Bit 0	ARE	Address Remap Enable: Set to ‘1’ to enable address remapping in order to access above 1 MByte range in Real Mode.

6.10.6. 32-BIT REAL MODE ACCESS

To access a memory mapped register having, for example, the address 004040F0h:

A memory access below 1 MByte (000030F0h) is generated by the CPU in Real Mode (the lower word, bits[11:0] should be the same as the actual address (0F0h). The steps involved are as follows:

– Program the Real Mode Aperture Register: Bits [19:12] to 03h

– Program the Remap Address Register to Upper WORD value (00404h)

– Program the Real Mode Control Register bit [0] to a ‘1’ to enable this feature

– Run any memory cycle to read/write to 000030F0h

6.11. ACCESSING CONFIGURATION REGISTERS

The Host interface and the Local Bus Interface are programmed identically. To access all the internal configuration registers, the programmer will need to program the Index (address) and data registers of the the required interface through port 22h/23h. The principle of the programming is to fix the address of device to a location that the user requires and to always adress it at that location. The steps required to access any of the internal registers are as follows:

1. Select the interface you want to programme. Each interface is described as a device and both have a number. The Local Bus device number is 6 and the Host device number is 7. The below example code describes how the devices are accessed and the Host device is used.

2. Select Host interface base programming option in RBI, by writing 0x00 and 0x07 (0x06 for the Local Bus, see Section 20.4. for more details) in register 0x11 (index value 0x11 accessed through I/O 22h/23h address) and 0x10 respectively.

```
IOWRITE8(0x22,0x10);

IOWRITE8(0x23,0x07);

IOWRITE8(0x22,0x11);

IOWRITE8(0x23,0x00);
```

3. Select the Host interface address. Assume that HOST_BASE is the address of the Host interface I/O space.

```
IOWRITE8(0x22,0x12);

IOWRITE8(0x23,(HOST_BASE &0xFF) | 0x03);

IOWRITE8(0x22,0x13);

IOWRITE8(0x23,HOST_BASE >>8);
```

The host interface registers are then accessed with HOST_BASE as the index register and HOST_BASE+4 as the data register, as shown below.

4. Writing into any Internal Register of the Host Interface:

```
IOWRITE8(HOST_BASE,offset);

IOWRITE32(HOST_BASE+4,data);
```

Here the “offset” index address is as mentioned in the register table shown above. The 32-bit “data” is written into the register.

5. Reading from any internal register of Host Input

```
IOWRITE8(HOST_BASE,offset);

IOREAD32(HOST_BASE+4,data);.
```

Here the “offset” index address is as mentioned in the register tables shown above. The 32-bit “data” is expected to be read from the internal register.

One constraint is that the Local Bus address must be set at multiples of 8h.

7. SDRAM CONTROLLER

7.1. INTRODUCTION

This chapter describes the mapping of the CPU memory and IO address spaces.

The STPC uses a Unified Memory Architecture; the system memory and the graphics buffers use the same memory space. This chapter provides information on the memory address map and the graphics memory usage, together with information on the arbitration logic which resolves accesses to the main memory. Details of memory shadowing and cachability by software control and the Memory Hole for ISA BIOS are also given. The actual interface to the external SDRAM modules is presented. Also introduced in this chapter are the PCI configuration space mapping registers, further details are in the chapter relating to the PCI Bus Controller.

7.2. MEMORY CONTROLLER

The STPC handles the memory data bus directly, controlling from 8 MBytes to 128 MBytes. The SDRAM controller supports accesses to the Memory Banks to/from the CPU (via the host), from the VIP, to/from the CRTC, and to the VIDEO with Video Pipeline & to/from the GE (Banks 0 to 3), and the local Bus which can be populated with either single-sided or double-sided 72-bit (4-bit parity) memory devices. Parity is not supported.

The SDRAM controller only supports 64 bit wide Memory Banks.

The SDRAM Controller supports buffered or unbuffered SDRAM but not EDO or FPM modes. SDRAMs must support Full Page Mode Type access.

The STPC Memory Controller provides various programmable SDRAM parameters to allow the SDRAM interface to be optimised for different processor bus speeds SDRAM speed grades and CAS Latency.

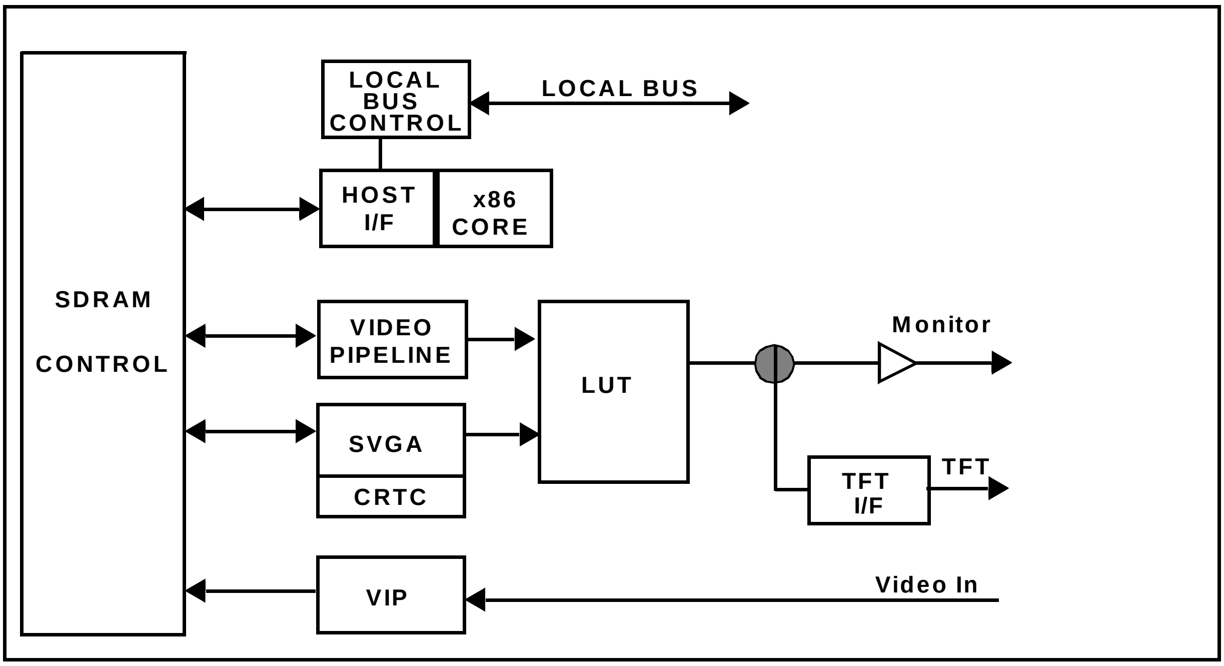


Figure 7-1. Memory Controller Interface Block Diagram

7.3. SDRAM REGISTER ACCESS

These registers are used to configure the SDRAM controller.

7.3.1. REGISTER 0

This is a 31-bit configuration register for the SDRAM controller block:

MEM_REG0 Access = Regoffset = 84C6000h

	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	RCASLCY			REGD	MRSR	RASo	LHDI	LGDI	LCDI	CASLat			Config		PRA
Default value after reset = 31x32198376h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRA	Rsv				RRW			BT	BL			RCT			
Default value after reset = 31x32198376h															

Bit Number	Mnemonic	Description
Bits 30-28	RCASLCY	**Read CAS Latency (RCASLCY)** should be equal to CASLCY but is left programmable for debug purpose.
Bit 27	REGD	**Registered DIMM**, Indicate if we use registered DIMMs ‘1’ or not ‘0’.
Bit 26	MRSR	**Mode Register Set Request,** If set to1, we update the SDRAM chips corresponds to the value programmed Bits [16:0].
Bit 25	RASo	**RAS on/off**, When we finish a read or a write, if set to 1 go into RACTIVE, if set to 0, go into PRECHARGE and the IDLE.
Bit 24	LHDI	**Latch_Host_Data_In** for host input data. If set to 0, select MD[63:0] directly from SDRAM, if set to 1, select lactched MD[63:0] by RDCLK (see bits 3-0 in Register 1).
Bit 23	LGDI	**Latch_GE_Data_In** for GE input data. If set to 0 it selects MD[63:0] directly from SDRAM , if set to 1 it selects latched MD[63:0] by RDCLK (see bits 3-0 in Register 1)
Bit 22	LCDI	**Latch_CRTC_Data_In** for CRTC input data. If set to 0, it select MD[63:0] directly from SDRAM, if set to 1 it selects latched MD[63:0] by RDCLK (see bits 3-0 in Register 1).
Bits 21-19	CASLat	**CAS Latency** This is abbreviated as CL in SDRAM datasheets and is defined as the number of clock cycles for the data held for after CAS goes low.

Bit Number	Mnemonic	Description
Bits 18-17	Config	**CONFIGURATION** of a DIMM. (See Table 7-1) These bits are used to determine the maximum burst length (full page burst). If the DIMMS are populated with a different kind of memory, the 2 bits are programmed to maximise the burst length using the minimum amount of the memory.
Bits 16-15	PRA	**Precharge to Row Active** number of cycles (called tRP in datasheets).
Bits 14-11	Rsv	**Reserved.** Should be set to ‘0000’.
Bits 10-8	RRW	**RACTIVE to Read/Write**(Called tRCD in datasheets).
Bit 7	BT	**Burst Type**.Should be set to ‘0’.
Bits 6-4	BL	**Burst Length**. Should be set to ‘111’
Bits 3-0	RCT	**Refresh Cycle Timing** (Called tRC in SDRAM datasheets).

Table 7-1. Memory Bank Configuration

Bit 18	Bit 17	Memory Bank Configuration	Maximum Burst Length
0	0	[4Mx4]x16	1024
0	1	[2Mx8]x8	512
1	0	[1Mx16]x4	256
1	1	Reserved	

7.3.2. REGISTER 1

This 7-bit register is used for the read clock scheme. See Chapter 6.3 “Clock considerations” for more details. This delay can be set up to 3.5 ns beyond the 15ns required from the previous MCLKI edge.

MEM_REG1 Access = Regoffset = 84C6004h

		5	4	3	2	1	0
		CSMEM	MEM	RCDP			
Default value after reset = 000000h							

Bit Number	Mnemonic	Description
Bit 5	CSMEM	**CS_MEM16_OE** It is programmed as bit 4 in case of 64Mbits and 128Mbits and is programmed to 1 otherwise.
Bit 4	MEM	**MEM16_OE_**, This bit is set to ‘1’ to get 16mA output enabled, set to ‘0’ to get 8mA output enabled
Bits 3-0	RCDP	**Read Clock Delay Programmation,** 0000 for smallest delay to 1111 for largest delay

7.3.3. REGISTER 2 (MEM_REG2) 84C6008H

This 2-bit register is used to determine the type of SDRAM in use.

MEM_REG2 Access = Regoffset = 84C6008h

Empty	1	0
Empty	SDRAM	
Default value after reset = 000000h		

Bit Number	Mnemonic	Description
Bits 1-0	SDRAM	**SDRAM type.** (See Table 7-2)

Table 7-2. SDRAM Type

Bit 1	Bit 0	Description
0	0	16 Mbits SDRAMs
0	1	64 Mbits or 128 Mbits 2 SDRAM banks
1	0	64 Mbits or 128 Mbits 4 SDRAM banks

7.4. MEMORY CLOCK REGISTERS

The MCLK register is used for Memory Clock control operations.

7.4.1. MCLK CONTROL REGISTER 0

MCLK00 Access = Regoffset = 0x40h

7	6	5	4	3	2	1	0
Uns	4-bitDIV				3-bitP		
Default value after reset = 0x5B							

Bit Number	Mnemonic	Description
Bit 7	Uns	Unused.
Bits 6- 3	4-bitDIV	This is the 4-bit M (divisor) value of the Memory synthesiser.
Bits 2-0	3-bitP	This is the 3-bit P (exponent) value of the Memory clock synthesizer.

This register defaults to 0x5B at reset. This value when combined with the default value of the other half of this pair results in a memory clock of 80.05 MHz assuming 14.318 MHz oscillator clock as the reference input.

For MCLK frequency programming values see table below:Table 7-3.

Note: If programming MCLK to a frequency not equal to HCLK, strap MD[5] must be disabled (i.e. set to 0) to remove the synchronization between these two clock signals (see Atlas Datasheet, STRAP OPTION section for further details).

$$\text{MCLK} = \frac{2 \times 14.31818 \times N}{M \times 2^P}$$

Constraints:

$$1 \leqslant M \leqslant 15$$
$$1 \leqslant N \leqslant 255$$
$$0 \leqslant P \leqslant 5$$

$$1\text{ MHz} \leqslant \frac{14.31818}{M} \leqslant 2\text{ MHz}$$

$$200\text{ MHz} \leqslant \frac{2 \times 14.31818 \times N}{M} \leqslant 622\text{ MHz}$$

7.4.2. MCLK CONTROL REGISTER 1

MCLK01			Access =			Regoffset = 0x41h	
7	6	5	4	3	2	1	0
8-bitN							
Default value after reset = 0xEC							

Bit Number	Mnemonic	Description
Bits 7- 0	8-bitN	These are bits 4-0 of the 8-bit N (multiplier) value of the Memory clock synthesiser.

This register defaults to 0xEC at reset. This value when combined with the default value of the other half of this pair results in a memory clock of 80.05 MHz assuming 14.318 MHz oscillator clock as the reference input.

For MCLK frequency programming values see Table 7-3

Table 7-3. MCLK Control Register Address 22 Index 40h, 41h

MHz	Reg1,Index 41h	Reg0, Index 40h	Actual Freq.	m	n	p
90	B0h	72h	89.999989	14	176	2
85	5Fh	42h	85.014194	8	95	2
80	7Bh	5Ah	80.051643	11	123	2
75	88h	6Ah	74.895095	13	136	2
66	53h	4Ah	66.022719	8	149	3
60	6Dh	6Ah	60.026216	13	109	2
55	A9h	5Bh	54.994828	11	169	3
50	4Dh	5Ah	50.113630	11	77	2
45	B0h	73h	44.999994	14	176	3
8	3Fh	3Dh	8.053978	7	63	5

7.5. SDRAM ARBITRATION:

Several agents, i.e. CPU, PCI masters, ISA masters, Graphics engine, CRT controller, Video Output, Video Input Port and Refresh controller, all compete for the system SDRAM memory.

A hierarchical arbitration scheme is used to optimise the SDRAM bandwidth usage. The system arbiter arbitrates among CPU, PCI and ISA masters. Refer to system arbiter section of this specification for details of how this is done. The winner of this arbitration, the system master, competes with the remaining agents for SDRAM. The SDRAM arbiter employs a dynamic arbitration algorithm to optimise the SDRAM utilization. The arbiter behaviour changes depending on whether the scan is close to and during the display of the video window:

- The following rules apply when the scan is not close to the video window.
- Refresh request is the lowest priority and is serviced only if no other agent is actively requesting.
- CRTC requests while current occupancy of the FIFO is above the low water mark are the next lowest priority requests and can be arbitrated out by GE, CRTC or video requests.
- CRTC requests when the occupancy is below the low water mark (urgent requests) have the highest priority will win over all other agents.
- Graphics engine requests lose to urgent CRTC and System master request. A System master request will terminate an ongoing Graphics service at the nearest CAS boundary while a CRTC request can terminate a on-going graphics service at the end of a sequence of read/write.
- The Video Output requests not close to the video window are prioritised just above the refresh.

When the scan is close to the video window and during the video window display, the arbiter behaviour changes significantly. The goal of the arbiter here is to ensure that the CRTC and Video FIFO occupancy is above a programmable minimum number of Bytes. This is necessary because, some memory and screen configurations do not have sufficient bandwidth availability. Since the drain rate is equal to the peak available bandwidth, it can not be sustained if all the pixels are to be fetched on demand. To overcome this, the arbiter ensures that a reservoir of CRTC and video pixels is available before the video window scan starts so that the difference of the fetch and drain rates can be made up for by dipping into this reservoir. This reservoir thus progressively shrinks as the video window is painted and approaches 0 Bytes by the end of the video window.

To ensure that the reservoir is filled up, a programmable distance before the video window x position, the arbiter switches over to a different set of low water marks for determining the urgency of the CRTC and video requests. Once urgent, these requests win over other requesters thus ensuring that the reservoir is full. Further, to avoid thrashing between CRTC and Video requests, the arbiter employs a programmable burst length to arbitrate between the two. Once the CRTC service is started, it is not interrupted by video until the burst length number of cycles have occurred and vice-versa. Since the drain rates of video changes with the scaling factor, the CRTC and video have different burst length parameter.

Once the video window repaints starts, the low water marks decrease linearly over the size of the window, to reflect the decreasing number of reservoir Bytes needed to make up for the difference in the fetch and drain rates. All other memory requesters are granted access, only if both CRTC and video FIFO occupancies are above their low water marks. The rules for granting the memory to the remaining agents are same as those listed above.

8. PCI CONTROLLERS

8.1. INTRODUCTION

PCI Specification 2.1, from PCI-SIG, for further details of the PCI bus standard.

The PCI bus is the main data communication link to the STPC chip. Four PCI devices are present internally in the STPC, a host to PCI bridge (North Bridge), a PCI to ISA bridge (South Bridge), an IDE controller and a USB controller. The STPC also contains a PCI arbiter which arbitrates between the bridges and for up to three external PCI devices. Figure 8-1 shows the layout of the PCI controllers within the STPC. Please refer to *PCI Specification 2.1*, from PCI-SIG, for further details of the PCI bus standard.

The *North Bridge* translates the appropriate host bus I/O and Memory cycles to the PCI bus. It also supports generation of Configuration cycles on the PCI bus. The Configuration Address register allows for the remapping of host CPU I/O cycles, in the address range 0xCF8h-0xCFFh, to configuration cycles on the PCI bus.

The North Bridge, as a PCI bus agent (host bridge class), fully complies with PCI specification 2.1. The North Bridge also implements the PCI mandatory header registers in Type 0 PCI configuration space for easy porting of PCI-aware system BIOS. The North Bridge is assigned the Device Number 0xBh, which corresponds to the IDSEL on AD11 signal. PCI configuration registers of the North Bridge are accessible by the Type 0 PCI configuration cycles generated at Device number 0xBh.

The South Bridge controller responds to PCI configuration read and write transactions. The South Bridge is assigned the Device Number 0xCh, which corresponds to the IDSEL on AD12 signal. PCI configuration registers of the South Bridge are accessible by the Type 0 PCI configuration cycles generated by the North Bridge.

The South Bridge, as a PCI bus agent (expansion bridge class), fully complies with PCI specification 2.1. The South Bridge implements PCI function 0, PCI to ISA bridge. As per PCI specification, the South Bridge will respond to function 0 configuration cycles directed to configuration slot 12 (AD12).

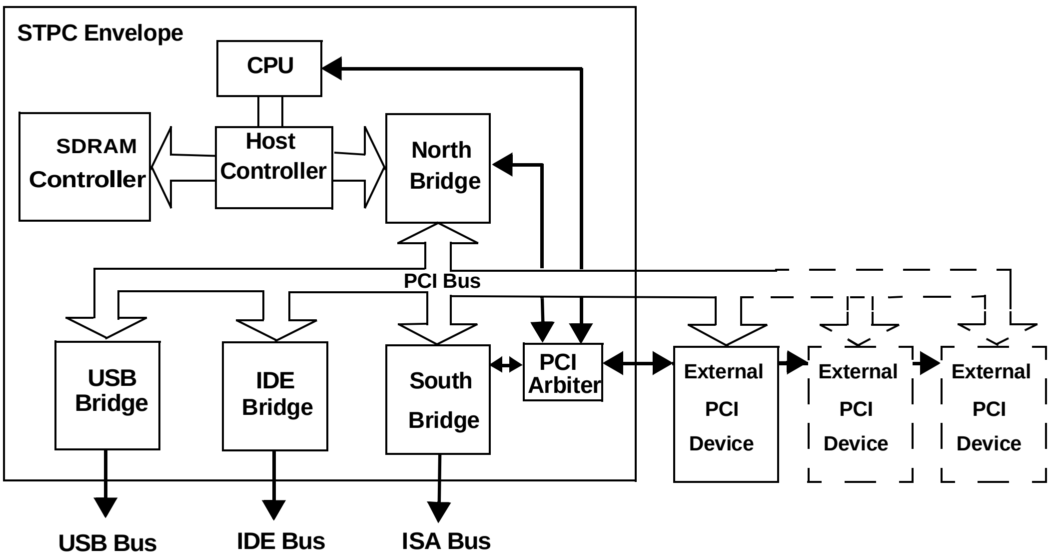


Figure 8-1. PCI Layout

8.1.1. PCI ADDRESS DECODE

The only positive decode carried out is for the IDE controller PIO registers. The decode address ranges are dictated by the IDE configuration registers, see IDE section for details. ISA resources are accessed only via subtractive decode.

8.1.2. PCI ERROR HANDLING

Under control of South Bridge configuration registers, one or more of the following events can generate a 1 PCICLK long pulse on SERR#, which in turn can be made to generate an NMI to the CPU.

ISA initiated transaction ending in target abort.

8.1.3. PCI ARBITER

The PCI arbiter controls access to the PCI bus when several bus masters are present in the system. Whenever a further potential bus master needs to gain access to the bus, it asserts its request. The arbiter then asserts a system hold condition, which eventually causes a hold signal to be asserted to the CPU. The CPU finishes the current instruction, tristates the internal bus and asserts a hold acknowledge. This eventually causes the assertion of a system hold acknowledge. Once the system hold acknowledge is asserted, the arbiter asserts a grant to whichever requesting master is in the front of the line in a round-robin chain. When there are no requests pending or when the CPU is requesting the bus and it is in the front of the line, control of the bus is passed back to the CPU by the negation of the system hold condition.

8.2. ACCESSING THE PCI CONFIGURATION REGISTERS

The PCI configuration registers are accessed, from the CPU, using two 32-bit registers, mapped as I/O at CF8h and CFCh.

Each read from and write to the PCI configuration registers must be done by:

Writing the 32-bit address of the PCI config. register using type 0 format at I/O CF8h.

Reading or Writing 32-bit data at CFCh

All PCI configuration registers, inside the North and South bridges and all other external PCI devices, are seen from the CPU through those 2 x 32-bit registers.

An illustration of these registers is shown in Table 8-1 & Table 8-2.

Table 8-1. Register CF8h

31	30 ----- 24	23 ------------ 16	15 -------------- 11	10 ------------------ 8	7 ------------------- 2	1 0
Enable	Reserved	Bus number	Device number	Function number	Register number	0

Table 8-2. Register CFCh

31 ---------------------------- 24	23 ---------------------------- 16	15 ----------------------------- 8	7 ------------------------------- 0
Byte 3	Byte 2	Byte 1	Byte 0

8.3. CONFIGURATION ADDRESS REGISTER

This is a 32-bit register accessible only via double-word IO read and write cycles.

Config_Address Access = 0xCF8h Regoffset =

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCI	Rsv							BN							
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DN					FN			RG						Rsv	
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bit 31	PCI	**PCI configuration register access enable.** When set to a ‘1’, host CPU I/O cycles in address range CFCh-CFFh are converted to configuration cycles on the PCI bus. Otherwise if set to a ‘0’, I/O cycles in this address range pass through as normal I/O cycles on the PCI bus.
Bits 30-24	Rsv	**Reserved.** Must be written to ‘0’. Read back as ‘0’.
Bits 23-16	BN	**Bus Number.** This field selects a specific bus number in the system. Bus Number 0 is assigned to the PCI bus directly behind the North Bridge. This field is driven on bits 23-16 of the AD bus during the address phase.
Bits 15-11	DN	**Device Number.** This field selects a specific device on the bus. During a Type-0 configuration cycle, this field is decoded to assert the appropriate IDSEL line as follows; The North Bridge Device Number 0xBh, which corresponds to IDSEL on AD11. The South Bridge Device Number 0xCh, which corresponds to IDSEL on AD12. IDE Bridge: Device Number 0xDh, which corresponds to IDSEL on AD13 USB Bridge: Device Number 0xEh, which corresponds to IDSEL on AD14
Bits 10-8	FN	**Function Number.** During a PCI configuration cycle, this field is driven on bits 10-8 of the AD bus of the PCI during the address phase.
Bits 7-2	RG	**Register Number.** During a PCI configuration cycle, this field is driven on bits 7-2 of the AD bus during the address phase.
Bits 1-0	Rsv	**Reserved**. Must be written to a ‘0’. Reads back as ‘0’.

8.4. CONFIGURATION DATA REGISTER

This is a 32-bit register accessible via 8-bit, 16-bit and 32-bit IO read and write cycles.

Config_Data Access = 0xCFCh Regoffset =

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

8.5. NORTH BRIDGE CONFIGURATION REGISTERS

The STPC North Bridge configuration registers are accessed using the values below :

Bus = 0h
Device = 0Bh (IDSEL internally connected to PCI address line 11)
Function = 0h (Host Bridge / PCI)

For example: Writing 80005800h at CF8h will access Vendor ID reg. index.

Table 8-3. North Bridge Reset Values

31 ---------------- 16		15 ---------------- 0		
Device ID : 020Ah		Vendor ID : 104Ah		00h
Status : 0280h		Command : 0007h		04h
Base class code: 06h	Sub class code: 00h	Program. Inter. Reg. : 00h	Revision : 00h	08h
	Header Type: 00h			0Ch
				...
Control Register : 00000000h				50h
Error Status Register : 00000000h				54h

8.5.1. NORTH BRIDGE PCI COMMAND REGISTER

This is the 16-bit PCI command register.

NB_Com Access = 0xCF8h/0xCFCh Regoffset = 0x4h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv							S	AD	P	VGA	MW	ES	BS	ME	IO E
Default value after reset = 0007h															

Bit Number	Mnemonic	Description
Bits 15-9	**Rsv**	**Reserved**. These bits are hardwired to ‘0’. Writes have no effect on them.
Bit 8	**S**	**SERR# enable.** If this bit is set to a ‘1’, the North Bridge may assert SERR# upon detecting a target abort in response to an North Bridge initiated PCI transaction, upon being forced to end a non-configuration space transaction with a master abort, or if a parity error on the PCI bus is detected. If this bit is set to ‘0’, the North Bridge will not assert SERR#.
Bit 7	**AD**	**Address/Data stepping enable.** This bit is hardwired to a ‘0’. Writes to it have no effect.
Bit 6	**P**	**PERR# response.** Must always be set to ’0’.
Bit 5	**VGA**	**VGA Palette Snoop enable.** This bit is hardwired to a ‘0’. Writes to it have no effect.
Bit 4	**MW**	**Master Write and Invalidate Enable.** This bit is hardwired to a ‘0’. Writes to it have have no effect.
Bit 3	**ES**	**Enable Special cycles.** This bit is hardwired to a ‘0’. Writes to it have no effect.
Bit 2	**BS**	**Bus Master enabled.** This bit is hardwired to a ‘1’. Writes to it have no effect.
Bit 1	**ME**	**Memory Enable.** This bit is hardwired to a ‘1’. Writes to it have no effect.
Bit 0	**I/O E**	**I/O Enable.** This bit is hardwired to a ‘1’. Writes to it have no effect.

8.5.2. NORTH BRIDGE PCI STATUS REGISTER

This is the 16-bit PCI Status register.

NB_Stat Access = 0xCF8h/0xCFCh Regoffset = 0x6h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DP	SS	SMA	RTA	STA	DT		DPED	FBBC	Rsv						
Default value after reset = 0280h															

Bit Number	Mnemonic	Description
Bit 15	**DP**	**Detected parity error.** This bit is set when a PCI parity error is detected. It may be cleared by software by writing a ‘1’ back to this bit.
Bit 14	**SS**	**Signalled SERR#.** This bit is set to a ‘1’ when SERR# is asserted by the North Bridge. Writing a ‘1’ to this bit will clear it.
Bit 13	**SMA**	**Signalled Master Abort.** This bit is set to a 1 when the North Bridge terminates a PCI transaction with a master abort. Writing a ‘1’ to this bit will clear it.
Bit 12	**RTA**	**Received Target Abort.** This bit is set to a ‘1’ when PCI transaction initiated by the North Bridge is terminated with a target abort. Writing a ‘1’ to this bit will clear it.
Bit 11	**STA**	**Signalled Target Abort.** This bit is hardwired to ‘0’.
Bits 10-9	**DT**	**DEVSEL Timing.** These bits are hardwired for medium timing to ‘01’. Writes have no effect.
Bit 8	**DPED**	**Data Parity Error Detected.** This bit is set to ‘1’ when a PCI data parity error is detected. Writing a ‘1’ will clear it.
Bit 7	**FBBC**	**Fast Back-to-Back Capable.** Hardwired to ‘1’. Indicates that the North Bridge, while acting as target, is capable of accepting fast back-to-back transactions. Reads will always return ‘1’, writes have no effect.
Bits 6-0	**Rsv**	**Reserved.** These bits are hardwired to ‘0’s.

8.5.3. NORTH BRIDGE PCI REVISION ID REGISTER

This is the 8-bit read only PCI revision identification register.

NB_R_ID Access = 0xCF8h/0xCFCh Regoffset = 0x8h

7	6	5	4	3	2	1	0
Rsv							
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-0	**Rsv**	**Reserved.** These bits are hardwired to 00h.

8.5.4. NORTH BRIDGE DEVICE CLASS CODE REGISTER

This is a 24-bit read only register implemented at configuration space offset 9h, Ah, Bh.

NB_C_Code Access = 0xCF8h/0xCFCh Regoffset = 0x9h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BCC								SCC							
Default value after reset = 06h								Default value after reset = 00h							

15	14	13	12	11	10	9	8
PIR							
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 31-24	**BCC**	**Base Class Code.** These bits are hardwired to 06h.
Bits 23-16	**SCC**	**Sub Class Code.** These bits are hardwired to 00h.
Bits 15-8	**PIR**	**Programming Interface Register.** These bits are hardwired to 00h.

8.5.5. NORTH BRIDGE HEADER TYPE REGISTER

This is an 8-bit read only register, hardwired to 00h

.

NB_Head Access = 0xCF8h/0xCFCh Regoffset = 0xEh

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0

8.5.6. NORTH BRIDGE CONTROL REGISTER

NB_Cont Access = 0xCF8h/0xCFCh Regoffset = 0x50h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv									PCI1	PCI2	PCI3	Rsv			
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv											P			SP	S
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-23	**Rsv**	**Reserved.** Hardwired to ‘0’.
Bit 22	**PCI1**	**PCI 2.0 Enable.** If this bit is set to ‘1’, North Bridge will be compatible with PCI 2.0 standard. If this bit is set to ‘0’, North Bridge is compatible with PCI 2.1 standard.
Bit 21	**PCI2**	**PCI to Host Read Prefetch Enable.** If this bit is set to ‘1’, all QWORD aligned burst reads from a PCI master addressed to the North Bridge system memory will use prefetch. If set to ‘0’, memory read cycles from PCI to host are allowed to complete before the PCI cycle is terminated and all burst read attempts will be disconnected on the PCI bus.
Bit 20	**PCI3**	**PCI to Host Write Posting Enable.** If this bit is set to ‘1’, all burst writes from a PCI master addressed to the North Bridge system memory will be posted. If it is set to ‘0’, all memory write cycles from PCI to host are allowed to complete before the PCI cycle is terminated and all burst write attempts will be disconnected on the PCI bus.
Bits 19-5	**Rsv**	**Reserved.** Hardwired to ‘0’.
Bit 4	**P**	**PERR_ on read data parity error enable.**
Bit 3	**P**	**PERR_ on write data parity error enable.**
Bit 2	**P**	**PERR_ on address parity error enable.**
Bit 1	**SP**	**SERR_ on PERR_ enable.**
Bit 0	**S**	**SERR_ on received target abort.**

8.5.7. NORTH BRIDGE PCI ERROR STATUS REGISTER

NB_E_Stat Access = 0xCF8h/0xCFCh Regoffset = 0x54h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv											RDP	WDP	AP	PES	RTAE
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-5	**Rsv**	**Reserved.** Hardwired to ‘0’.
Bit 4	**RDP**	**Read Data Parity Error Status.** This bit is set when a PCI read data parity error is detected. Writing a ‘1’ will clear it.
Bit 3	**WDP**	**Write Data Parity Error Status.** This bit is set when a PCI write data parity error is detected. Writing a ‘1’ will clear it.
Bit 2	**AP**	**Address Parity Error Status.** This bit is set when a PCI address parity error is detected. Writing a ‘1’ will clear it.
Bit 1	**PES**	**Parity Error Status.** System errors as a result of a parity error status. This bit is set to ‘1’ when SERR# was asserted as a result of parity error. Writing a ‘1’ will clear it.
Bit 0	**RTAE**	**Received Target Abort Error.** System errors as a result of a received target abort. This bit is set to ‘1’ when SERR# was asserted as a result of receiving a target abort. Writing a ‘1’ will clear it.

8.6. THE SOUTH BRIDGE

The STPC South Bridge configuration registers are accessed using the values below :

Bus = 0
Device = Ch (IDSEL internally connected to PCI address line 12)
Function = 0 (ISA bridge)
- Responds to I/O / memory / config
- Translates Master ISA to PCI
- Translates PCI to Slave ISA

8.7. SOUTH BRIDGE PCI PCI to ISA CONFIGURATION REGISTERS

Table 8-4. ISA Bridge Configuration Space Register Reset Values

31	16	15	0	
Device ID: 0210h		Vendor ID: 104Ah		00h
Status: 0280h		Command: 000Fh		04h
Base class code: 06h	Sub class code: 01h	Program. Inter. Reg. : 00h	Revision ID: 00h	08h
	Header: 40h			0Ch
				...
				...
			Miscellaneous reg : 00h	40h

This section describes Function 0 (F#0) configuration registers, including the PCI to ISA bridge control. The registers and reset values are illustrated in Table 8-4.

8.7.1. SOUTH BRIDGE PCI COMMAND REGISTER

This is the 16-bit PCI command register.

SB_Com_0 Access = 0xCF8h/0xCFCh Regoffset = 0x4h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv							S	AD	P	VGA	MW	ESC	BM	ME	IO E
Default value after reset = 000Fh															

Bit Number	Mnemonic	Description
Bits 15-9	**Rsv**	**Reserved**. These bits are hardwired to ‘0’s. Writes have no effect on them.
Bit 8	**S**	**SERR# enable.** If this bit is set to a ‘1’, the South Bridge may assert SERR# upon detecting a target abort in response to a South Bridge initiated PCI transaction on behalf of an ISA master. If this bit is set to ‘0’, the South Bridge will not assert SERR#.
Bit 7	**AD**	**Address/Data stepping enable.** This bit is hardwired to a ‘0’. Writes to it have no effect.
Bit 6	**P**	**PERR# response.** Setting this bit to ‘1’ enables parity error detection.
Bit 5	**VGA**	**VGA Palette Snoop enable.** This bits is hardwired to a ‘0’. Writes to it have no effect.
Bit 4	**MW**	**Master Write and Invalidate Enable.** This bit is hardwired to a ‘0’. Writes to it have no effect.
Bit 3	**ESC**	**Enable Special Cycles.** This bit is hardwired to a ‘1’. The South Bridge writes to it have no effect. The South Bridge responds to halt and shutdown cycles.
Bit 2	**BM**	**Bus Master enabled.** This bit is hardwired to a ‘1’. Writes to it have no effect.
Bit 1	**ME**	**Mem Enable.** This bit is hardwired to a ‘1’. Writes to it have no effect.
Bit 0	**I/O E**	**I/O Enable.** This bit is hardwired to a ‘1’. Writes to it have no effect.

8.7.2. SOUTH BRIDGE PCI STATUS REGISTER

This is the 16-bit PCI Status register.

SB_Stat0 Access = 0xCF8h/0xCFCh Regoffset = 0x6h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv	SS	SMA	RTA	STA	DT		DPED	FBBC	Rsv						
Default value after reset = 0280h															

Bit Number	Mnemonic	Description
Bit 15	**Rsv**	**Reserved**. This bit is hardwired to ‘0’.
Bit 14	**SS**	**Signalled SERR#.** This bit is set to a ‘1’ when SERR# is asserted by the South Bridge on behalf of an ISA master cycle. Writing a ‘1’ to this bit will clear it. SERR# is asserted in response to a target abort during an ISA master cycle on PCI bus and if bit-8 of the F#0 PCI command register is set to a ‘1’ to enable SERR# signalling.
Bit 13	**SMA**	**Signalled Master Abort.** This bit is hardwired to a ‘0’.
Bit 12	**RTA**	**Received Target Abort.** This bit is set to a ‘1’ when the PCI transaction is initiated by the South Bridge on behalf of an ISA master is terminated with a target abort. Writing a ‘1’ to this bit will clear it.
Bit 11	**STA**	**Signalled Target Abort.** This bit is set to a ‘1’ when the South Bridge terminates a PCI transaction with a target abort. Writing a ‘1’ to this bit will clear it. The South Bridge will generate target abort if a A1-0 of a PCI IO cycle does not match the Byte enables.
Bits 10-9	**DT**	**DEVSEL Timing.** These bits are hardwired for medium timing to ‘01’. Writes have no effect.
Bit 8	**DPED**	**Data Parity Error Detected.** This bit is hardwired to ‘0’.
Bit 7	**FBBC**	**Fast Back-to-Back Capable.** Hardwired to ‘1’. Indicates that the South Bridge, while acting as target, is capable of accepting fast back-to-back transactions. Reads will always return ‘1’, writes have no effect.
Bits 6-0	**Rsv**	**Reserved.** These bits are hardwired to ‘0’.

8.7.3. SOUTH BRIDGE PCI REVISION ID REGISTER

This is the 8-bit read only PCI revision identification register.

SB_R_ID0 Access = 0xCF8h/0xCFCh Regoffset = 0x8h

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0

Bits 7-0: These bits are hardwired to 00h.

8.7.4. SOUTH BRIDGE DEVICE CLASS CODE REGISTER

This is a 24-bit read only register implemented at configuration space offset 09h, 0Ah, 0Bh.

SB_C_Code0 Access = 0xCF8h/0xCFCh Regoffset = 0x9h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BCC								SCC							
Default value after reset = 06h								Default value after reset = 01h							

15	14	13	12	11	10	9	8
PIR							
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 31-24	**BCC**	**Base Class Code.** These bits are hardwired to 06h (Bridge Device).
Bits 23-16	**SCC**	**Sub Class Code.** These bits are hardwired to 01h (ISA Bridge).
Bits 15-8	**PIR**	**Programming Interface Register.** These bits are hardwired to 00h.

8.7.5. SOUTH BRIDGE HEADER TYPE REGISTER

This register is hardwired to 80h, indicating that the South Bridge is a multi-function PCI device.

SB_Head0 Access = 0xCF8h/0xCFCh Regoffset = 0xEh

7	6	5	4	3	2	1	0
1	0	0	0	0	0	0	0

8.7.6. SOUTH BRIDGE MISCELLANEOUS REGISTER

SB_Misc0			Access = 0xCF8h/0xCFCh				Regoffset = 040h
7	6	5	4	3	2	1	0
Rsv				GPIO_MDC	GPIO_SDC	GPIO	PCI
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-4	**Rsv**	**Reserved.** Hardwired to 00h.
Bit 3	**GPIO_MDC**	**GPIO Master Debounce Control :** enable/disable the debounce logic on master GPIO 0 = Disabled 1 = Enabled
Bit 2	**GPIO_SDC**	**GPIO Slave Debounce Control**: enable/disable the debounce logic on slave GPIO 0 = Disabled 1 = Enabled
Bits 1	**GPIO**	GPIO[1)] Enable/Disable: enable/disable all GPIOs 0 = Disabled 1 = Enabled
Bit 0	**PCI**	**PCI 2.0 Enable.** If this bit is set to ‘1’, South Bridge will be compatible with PCI 2.0 standard. If this bit is set to ‘0’, South Bridge is compatible with PCI 2.1 standard.

Note 1. Enabling GPIO will enable both master and slave GPIO controllers.

Programming Notes; This register is used to initiallise the GPIOs as well as select the mode in which the Bridge operates. For information on how to initialise the GPIOs, refer to Table 21-1

8.8. PCI to IDE BRIDGE CONFIGURATION REGISTERS

This section describes the PCI to IDE Bridge configuration registers. The registers and reset values are illustrated in Table 8-5 Table 6-12.

The PCI to IDE Bridge configuration registers are accessed using the values below:

Bus = 0

Device = 0Dh (IDSEL internally connected to PCI address line 13)

Function = 0(IDE controller)

- Responds to IO / config

For example: Writing 80006800h at CF8h will access Function 0 (IDE) Command reg. index.

Table 8-5. PCI to IDE Bridge Configuration Space Register Reset Values

<table>
<tr><td colspan="2">31 … 16</td><td colspan="2">15 … 0</td><td></td></tr>
<tr><td colspan="2">Device: 0228h</td><td colspan="2">Vendor ID: 104Ah</td><td>00h</td></tr>
<tr><td colspan="2">Status: 0280h</td><td colspan="2">Command: 0000h</td><td>04h</td></tr>
<tr><td>Base class code: 01h</td><td>Sub class code: 01h</td><td>Program. Inter. Reg: 8Ah</td><td>Revision ID: 00h</td><td>08h</td></tr>
<tr><td></td><td>Header: 40h</td><td>Reserved: 00h</td><td></td><td>0Ch</td></tr>
<tr><td colspan="4">IO Base address 0 register: 00000001h</td><td>10h</td></tr>
<tr><td colspan="4">IO Base address 1 register: 00000001h</td><td>14h</td></tr>
<tr><td colspan="4">IO Base address 2 register: 00000001h</td><td>18h</td></tr>
<tr><td colspan="4">IO Base address 3 register: 00000001h</td><td>1Ch</td></tr>
<tr><td colspan="4">Reserved</td><td>20h</td></tr>
<tr><td colspan="4"></td><td>...
...</td></tr>
<tr><td colspan="4">Primary IDE Timing register: 97609760h</td><td>40h</td></tr>
<tr><td colspan="4">Secondary IDE Timing register: 97609760h</td><td>44h</td></tr>
<tr><td colspan="3"></td><td>Miscellaneous reg : 00h</td><td>48h</td></tr>
</table>

8.8.1. PCI to IDE BRIDGE VENDOR IDENTIFICATION REGISTER

This is a 16-bit read-only register implemented at configuration space offset 00h and 01h. It contains the Vendor Identifier assigned to STPC.

Bits 15-0 These bits are hardwired to 104Ah.

IDEB_V_ID1 Access = 0xCF8h/0xCFCh Regoffset = 0x0h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	0	0	0	1	0	0	1	0	1	0

Writes to this register have no effect.

8.8.2. PCI to IDE BRIDGE DEVICE IDENTIFICATION REGISTER

This is a 16-bit read only register implemented at configuration space offset 02h and 03h. It contains the Device Identifier assigned to the PCI to EIDE bridge.

Bits 15-0 These bits are hardwired to 0228h

IDEB_D_ID1 Access = 0xCF8h/0xCFCh Regoffset = 0x2h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0

Writes to this register have no effect.

8.8.3. PCI to IDE BRIDGE PCI COMMAND REGISTER

This is the 16-bit PCI command register.

IDEB_Com1 Access = 0xCF8h/0xCFCh Regoffset = 0x4h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv							SE	A	P	VGA	MWIE	ESC	BM	M E	IO E
Default value after reset = 0005h															

Bit Number	Mnemonic	Description
Bits 15-9	**Rsv**	**Reserved.** These bits are hardwired to ‘0’s. Writes have no effect on them.
Bit 8	**SE**	**SERR# Enable.** If this bit is set to a ‘1’, the PCI to IDE Bridge may assert SERR# upon detecting a master or target abort in response to a the PCI to IDE Bridge initiated PCI transaction on behalf of IDE master. If this bit is set to ‘0’, the PCI to IDE Bridge will not assert SERR#.
Bit 7	**A**	**Address/Data stepping enable.** This bit is hardwired to a ‘0’. Writes to it have no effect.
Bit 6	**P**	**PERR# response.** Setting this bit to ‘1’ enables parity error detection.
Bit 5	**VGA**	**VGA Palette Snoop enable.** This bits is hardwired to a ‘0’. Writes to it have no effect.
Bit 4	**MWIE**	**Master Write and Invalidate Enable.** This bit is hardwired to a ‘0’. Writes to it have no effect.
Bit 3	**ESC**	**Enable Special cycles.** This bit is hardwired to a ‘0’. Writes to it have no effect.
Bit 2	**BM**	**Bus Master enabled.** This bit is hardwired to a ‘1’. Writes to it have no effect.
Bit 1	**M E**	**Mem Enable.** This bit is hardwired to a ‘0’. Writes to it have no effect.
Bit 0	**IO E**	**IO Enable.** Setting this bit to a ‘1’ enables access to the IDE IO registers.

8.8.4. PCI to IDE BRIDGE PCI STATUS REGISTER

IDEB_Stat1 Access = 0xCF8h/0xCFCh Regoffset = 0x6h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv	SS	SMA	RTA	Rsv	DT		DPED	FBBC	Rsv						
Default value after reset = 0280h															

Bit Number	Mnemonic	Description
Bit 15	**Rsv**	**Reserved**. This bit is hardwired to ‘0’.
Bit 14	**SS**	**Signalled SERR#.** This bit is set to a ‘1’.
Bit 13	**SMA**	**Signalled Master Abort.** This bit is set to a ‘1’ when the PCI to IDE bridge terminates a PCI transaction initiated on behalf of the IDE master with a master abort. The PCI to IDE bridge master aborts an IDE master cycle if no target responds to this cycle.
Bit 12	**RTA**	**Received Target Abort.** This bit is set to a ‘1’ when a PCI transaction initiated by the PCI to IDE bridge on behalf of the IDE master is terminated with a target abort. Writing a ‘1’ to this bit will clear it.
Bit 11	**Rsv**	**Reserved**. This bit is hardwired to ‘0’.
Bits 10-9	**DT**	**DEVSEL Timing.** These bits are hardwired for medium timing to ‘01’. Writes to these bits have no effect.
Bit 8	**DPED**	**Data Parity Error Detected.** This bit is hardwired to ‘0’.
Bit 7	**FBBC**	**Fast Back-to-Back Capable.** This bit is hardwired to ‘1’.
Bits 6-0	**Rsv**	**Reserved**. These bits are hardwired to ‘0’.

8.8.5. PCI to IDE BRIDGE REVISION ID REGISTER

This is the 8-bit read only PCI revision identification register.

IDEB_R_ID1 Access = 0xCF8h/0xCFCh Regoffset = 0x8h

7	6	5	4	3	2	1	0
Rsv							
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-0	**Rsv**	**Reserved.** These bits are hardwired to 00h in this stepping of the chip.

8.8.6. PCI to IDE BRIDGE PROGRAMMING INTERFACE REGISTER

Prog_Int Access = 0xCF8h/0xCFCh Regoffset = 0x09h

7	6	5	4	3	2	1	0
Rsv	Rsv			SCPS	OPMODE2	SCPP	OPMODE1
Default value after reset = 8Ah							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved.** This bit is hardwired to ‘1’. Writes to have no effect on this bit.
Bits 6-4	**Rsv**	**Reserved.** These bits are hardwired to ‘0’. These bits are hardwired to ‘0’.
Bit 3	**SCPS**	This bit is hardwired to ‘1’ indicating that the secondary channel is programmable to be either in legacy or native mode.
Bit 2	**OPMODE2**	This bit selects the operating mode of the secondary channel. (see Table 8-6)
Bit 1	**SCPP**	This bit is hardwired to ‘1’ indicating that the primary channel is programmable to be either in legacy or native mode.
Bit 0	**OPMODE1**	This bit selects the operating mode of the primary channel. (see Table 8-7)

Table 8-6. Operating Mode of the Secondary Channel

Bit 2	
0	Channel is in legacy mode. In legacy mode the secondary IDE channel occupies IO addresses 170h-177h and 376h.
1	Channel is in native mode. The address range occupied by the secondary IDE controller in native mode is specified by base address registers 2 and 3.

Table 8-7. Operating Mode of the Primary Channel

Bit 0	
0	Channel is in legacy mode. In legacy mode the Primary IDE channel occupies IO addresses 1F0h-1F7h and 3F6h.
1	Channel is in native mode. The address range occupied by the Primary IDE controller in native mode is specified by base address registers ‘0’ and ‘1’.

8.8.7. PCI to IDE BRIDGE SUB-CLASS CODE REGISTER

This register is hardwired to 01h indicating that this is an IDE controller device.

Sub_Class Access = 0xCF8h/0xCFCh Regoffset = 0xAh

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1

8.8.8. PCI to IDE BRIDGE BASE-CLASS CODE REGISTER

This register is hardwired to 01h indicating that Function 1 is a mass storage device.

Base_Class Access = 0xCF8h/0xCFCh Regoffset = 0xBh

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1

8.8.9. PCI to IDE BRIDGE LATENCY TIMER CONTROL REGISTER

Lat_T Access = 0xCF8h/0xCFCh Regoffset = 0xDh

7	6	5	4	3	2	1	0
Rsv							
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-0	**Rsv**	**Reserved.** These bits are hardwired to ‘0’.

8.8.10. PCI to IDE BRIDGE HEADER TYPE REGISTER

This register is hardwired to 40h indicating that the PCI to IDE Bridge is a PCI multi-function device.

Head_T Access = 0xCF8h/0xCFCh Regoffset = 0xEh

7	6	5	4	3	2	1	0
0	1	0	0	0	0	0	0

8.8.11. PCI to IDE BRIDGE IDE BASE ADDRESS 0 REGISTER

This 32-bit register contains the base IO address for accessing the primary IDE channel's command registers. The base address is meaningful only when the Primary channel is programmed for native mode operation. If programmed for legacy mode operation, the primary channel's command registers are decoded at 1F0h IO address.

Base0 Access = 0xCF8h/0xCFCh Regoffset = 0x10h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BA															
Default value after reset = 00000001h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BA													Rsv	Rsv	MSI
Default value after reset = 00000001h															

Bit Number	Mnemonic	Description
Bits 31-3	**BA**	**Base Address.** This field specifies the 8-Byte IO address range where the primary channel command registers are located.
Bit 2	**Rsv**	Hardwired to '0' to indicate that this base address occupies 4-Bytes in IO space.
Bit 1	**Rsv**	**Reserved**. Hardwired to '0'.
Bit 0	**MSI**	**Memory Space Indicator.** This bit is hardwired to '1' to indicate IO space.

8.8.12. PCI to IDE BRIDGE IDE BASE ADDRESS 1 REGISTER

This 32-bit register contains the base IO address for accessing the primary IDE channel's Control registers. The base address is meaningful only when the Primary channel is programmed for native mode operation. If programmed for legacy mode operation, the primary channel's control register are decoded at 3F6h.

Base1 Access = 0xCF8h/0xCFCh Regoffset = 0x14h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BA															
Default value after reset = 00000001h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BA														Rsv	MSI
Default value after reset = 00000001h															

Bit Number	Mnemonic	Description
Bits 31-2	**BA**	**Base Address.** This field specifies the 4-Byte IO address range where the primary channel command registers are located.
Bit 1	**Rsv**	**Reserved**. Hardwired to '0'.
Bit 0	**MSI**	**Memory Space Indicator.** This bit is hardwired to '1' to indicate IO space.

8.8.13. PCI to IDE BRIDGE IDE BASE ADDRESS 2 REGISTER

This 32-bit register contains the base IO address for accessing the secondary IDE channel's command registers. The base address is meaningful only when the secondary channel is programmed for native mode operation. If programmed for legacy mode operation, the secondary channel's command registers are decoded at 170h IO address.

Base2 Access = 0xCF8h/0xCFCh Regoffset = 0x18h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BA															
Default value after reset = 00000001h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BA													Rsv	Rsv	MSI
Default value after reset = 00000001h															

Bit Number	Mnemonic	Description
Bits 31-3	**BA**	**Base Address.** This field specifies the 8-Byte IO address range where the secondary channel command registers are located.
Bit 2	**Rsv**	Hardwired to '0' to indicate that this base address occupies 4-Bytes in IO space.
Bit 1	**Rsv**	**Reserved**. Hardwired to '0'.
Bit 0	**MSI**	**Memory Space Indicator.** This bit is hardwired to '1' to indicate IO space.

8.8.14. PCI to IDE BRIDGE IDE BASE ADDRESS 3 REGISTER

This 32-bit register contains the base IO address for accessing the secondary IDE channel's Control registers. The base address is meaningful only when the secondary channel is programmed for native mode operation. If programmed for legacy mode operation, the secondary channel's control register is decoded at 376h.

Base3 Access = 0xCF8h/0xCFCh Regoffset = 0x1Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BA															
Default value after reset = 00000001h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BA														Rsv	MSI
Default value after reset = 00000001h															

Bit Number	Mnemonic	Description
Bits 31-2	**BA**	**Base Address.** This field specifies the 4-Byte IO address range where the secondary channel command registers are located.
Bit 1	**Rsv**	**Reserved**. Hardwired to ‘0’.
Bit 0	**MSI**	**Memory Space Indicator.** This bit is hardwired to ‘1’ to indicate IO space.

8.8.15. PCI to IDE BRIDGE IDE BASE ADDRESS 4 REGISTER

This 32-bit register contains the base IO address for accessing the bus master control and status register.

Base4 Access = 0xCF8h/0xCFCh Regoffset = 0x20h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BA															
Default value after reset = 00000001h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												HW		Rsv	MSI
Default value after reset = 00000001h															

Bit Number	Mnemonic	Description
Bits 31-4	**BA**	**Base Address.** This field specifies the 16-bytes IO address range where the Bus master control and status registers are located.
Bits 3-2	**HW**	Hardwired to ‘0’ to indicate that this base address occupies 16-bytes in IO space.
Bit 1	**Rsv**	**Reserved**. Hardwired to ’0’.
Bit 0	**MSI**	**Memory space indicator.** This bit is hardwired to ‘1’ to indicate IO space.

8.8.16. PCI to IDE BRIDGE IDE TIMING REGISTER

This 16-bit register contains all the IDE timing information for the Read and Write signals. This register is duplicated with the appropriate offsets for the Primary Master, Slave, Secondary Master or Slave Timings. The offsets are shown in Table 8-8

Table 8-8. Timing Register Location

Regoffset	Timing Register
040h	Primary Master Timing Control
042h	Primary Slave Timing Control
044h	Secondary Master Timing Control
046h	Secondary Slave Timing Control

IDE_Timing Access = 0xCF8h/0xCFCh Regoffset = Table 8-8

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDSMS		IDRT		IDAT		IPRT			IPAT			ISE	EWP	ERP	EPA
Default value after reset = 7F60h															

Bit Number	Mnemonic	Description
Bits 15-14	**IDSMS**	**IDE DMA Speed Mode Select.** These bits, along with bits 29-26, determine the width of the read and write signals during DMA transfers. Refer to Table 8-9 to determine the number of clocks for active and recovery times for the read/write signals. The slower modes are normally used for single word DMA modes and the faster modes are used for double word DMA modes.
Bits13-12	**IDRT**	**IDE DMA Recovery Time.** These bits, along with bits 31-30, determine the duration of the recovery (inactive) time of the read/write signals during DMA transfers. Refer to Table 8-10 to determine the number of clocks for recovery times of the read/write signals. Default is 01h.
Bits 11-10	**IDAT**	**IDE DMA Active Time.** These bits, along with bits 31-30, determine the duration of the active time of the read/write signals during DMA transfers. Refer to Table 8-11 to determine the number of clocks for active times of the read/write signals. Default is 01h.
Bits 9-7	**IPRT**	**IDE PIO Recovery Time.** These bits determine the duration of the recovery (inactive) time of the read/write signals during PIO data transfers. The transfers to non-data registers of the IDE device will always have a recovery time of 10 clocks. Refer to Table 8-12 to determine the number of clocks for recovery times of the read/write signals.

Bit Number	Mnemonic	Description
Bits 6-4	IPAT	**IDE PIO Active Time.** These bits determine the duration of the active time of the read/write signals during PIO transfers. The transfers to non-data registers of the IDE device will always have an active time of 10 clocks. Refer to Table 8-13 to determine the number of clocks for active times of the read/write signals: Note that the address setup time is implied. After initial start-up latency, the address setup time is given in Table 8-14
Bit 3	ISE	**IOCHRDY Sampling Enable.** This bit when set to 1, enables IOCHRDY sampling of the IDE device, i.e., the active read/write signals will be stretched (when IOCHRDY is low), to extend the cycle, if this bit is set. When set to 0, IOCHRDY is ignored and the read/write signals are deasserted after the specified number of PCI clocks.
Bit 2	EWP	**Enable Write Posting.** This bit when set to 1, enables posting of data into the FIFO during PIO data write transfers. Note that the non-data writes are never posted.
Bit 1	ERP	**Enable Read Prefetch.** This bit when set to 1 enables data to be prefetched into the data FIFO during PIO read commands. If set to 0, prefetch is completely disabled.
Bit 0	EPA	**Enable Prefetch for ATAPI commands.** When set to 1, this bit enables prefetching for ATAPI commands (when A0h is written into the command register - Register offset 07h for the device). When set to 0, prefetching during ATAPI commands is disabled. This is to accommodate non-512 boundary data transfers that are supported by ATAPI devices. Prefetch for the IDE controller is given in Table 8-15.

Table 8-9. DMA Speed Mode Select

Bit 15	Bit 14	DMA Speed Mode
0	0	Fast mode
0	1	Medium fast mode
1	0	Medium slow
1	1	Slow mode

Table 8-10. IDE DMA Recovery Time Settings

		Bit 13-12	Bit 13-12	Bit 13-12	Bit 13-12
Bit 15	Bit 14	00	01	10	11
0	0	1	2	3	4
0	1	5	6	7	8
1	0	9	10	11	12
1	1	14	15	16	20

Table 8-11. IDE DMA Active Time Settings

	Bit 11-10	Bit 11-10	Bit 11-10	Bit 11-10
Bit 15-14	00	01	10	11
00	1	2	3	4
01	5	6	7	8
10	9	10	11	12
11	14	15	16	20

Table 8-12. Recovery R/W Signal Time

Bit 9	Bit 8	Bit 7	PCI Clocks
0	0	0	1
0	0	1	2
0	1	0	3
0	1	1	4
1	0	0	8
1	0	1	9
1	1	0	10
1	1	1	12

Table 8-13. Active R/W Signal Time

Bit 6	Bit 5	Bit 4	PCI Clocks
0	0	0	2
0	0	1	3
0	1	0	4
0	1	1	5
1	0	0	8
1	0	1	9
1	1	0	10
1	1	1	12

Table 8-14. Address Setup Time

Bit 3	Bit 2	Address Setup Time
0	0	1 clock
0	1	2 clocks
1	X	3 clocks

Table 8-15. Prefetch Encoding

Bit 1	Bit 0	Prefetch
0	X	Completely disabled
1	0	Enabled for non-ATAPI commands only
1	1	Enabled for all commands

8.8.17. PCI to IDE BRIDGE MISCELLANEOUS REGISTER

This register contains miscellaneous information.

IDEB_Misc1 Access = 0xCF8h/0xCFCh Regoffset = 0x48h

7	6	5	4	3	2	1	0
SR	Rsv			DSC	DPC	SID	PID
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**SR**	**Soft Reset.** When set to 1, the IDE controller is reset. It does not affect the timing control register. The FIFOs and the internal state machines are cleared.
Bits 6-4	**Rsv**	**Reserved.**
Bit 3	**DSC**	**Disable Secondary channel.** When set to 1, the secondary channel interrupts and accesses to the secondary channel IDE command and control registers are disabled.
Bit 2	**DPC**	**Disable Primary channel.** When set to 1, the primary channel interrupts and accesses to the primary channel IDE command and control registers are disabled.
Bit 1	**SID**	**Secondary Interrupt Detect.** This bit is set when the secondary interrupt is active. It is cleared by writing a 1 to this bit in the register.
Bit 0	**PID**	**Primary Interrupt Detect.** This bit is set when the primary interrupt is active. It is cleared by writing a 1 to this bit in the register.

8.9. PCI to USB BRIDGE CONFIGURATION REGISTERS

The PCI to USB Bridge configuration registers are accessed using the values below:

Bus = 0

Device = 0Eh (IDSEL internally connected to PCI address line 14)

Function = 0 (USB controller)

- Responds to IO / config

For example: Writing 80007000h at CF8h will access Function 0 (USB) Command reg. index 0h.

Table 8-16. PCI to USB Bridge Configuration Space Register Reset Values

31	16	15	0	
Device: 0230h		Vendor ID: 104Ah		00h
Status: 0280h		Command: 0000h		04h
Base class code: 0Ch	Sub class code: 03h	Program. Inter. Reg.: 10h	Revision ID: 00h	08h
	Header: 40h	Reserved: 00h		0Ch
Command				10h
Class_Code				14h
BAR_OHCI				18h
				1Ch
				20h

8.9.1. PCI to USB BRIDGE VENDOR IDENTIFICATION REGISTER

This is a 16-bit read-only register implemented at configuration space offset 00h and 01h. It contains the Vendor Identifier assigned to STPC.

Bits 15-0 These bits are hardwired to 104Ah.

USBB_V_ID1 Access = 0xCF8h/0xCFCh Regoffset = 0x0h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	0	0	0	0	1	0	0	1	0	1	0

Writes to this register have no effect.

8.9.2. PCI to USB BRIDGE DEVICE IDENTIFICATION REGISTER

This is a 16-bit read only register implemented at configuration space offset 02h and 03h. It contains the Device Identifier assigned to the PCI to USB bridge.

Bits 15-0 These bits are hardwired to 0230h

USBB_D_ID1 Access = 0xCF8h/0xCFCh Regoffset = 0x2h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0

Writes to this register have no effect.

8.9.3. USB BRIDGE PCI COMMAND REGISTER

This is the 16-bit PCI command register.

USBB_Com Access = 0xCF8h/0xCFCh Regoffset = 0x4h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv							S	AD	P	VGA	MW	ES	BS	ME	IO E
Default value after reset = 0006h															

Bit Number	Mnemonic	Description
Bits 15-9	**Rsv**	**Reserved**. These bits are hardwired to ‘0’. Writes have no effect on them.
Bit 8	**S**	**SERR# enable.** If this bit is set to a ‘1’, the USB Bridge may assert SERR# upon detecting a target abort in response to a USB Bridge initiated PCI transaction, upon being forced to end a non-configuration space transaction with a master abort, or if a parity error on the PCI bus is detected. If this bit is set to ‘0’, the USB Bridge will not assert SERR#.
Bit 7	**AD**	**Address/Data stepping enable.** This bit is hardwired to a ‘0’. Writes to it have no effect.
Bit 6	**P**	**PERR# response.** Must always be set to ’0’.
Bit 5	**VGA**	**VGA Palette Snoop enable.** This bit is hardwired to a ‘0’. Writes to it have no effect.
Bit 4	**MW**	**Master Write and Invalidate Enable.** This bit is hardwired to a ‘0’. Writes to it have no effect.
Bit 3	**ES**	**Enable Special cycles.** This bit is hardwired to a ‘0’. Writes to it have no effect.
Bit 2	**BS**	**Bus Master enabled.** This bit is hardwired to a ‘1’. Writes to it have no effect.
Bit 1	**ME**	**Memory Enable.** This bit is hardwired to a ‘1’. Writes to it have no effect.
Bit 0	**I/O E**	**I/O Enable.** This bit is hardwired to a ‘0’. Writes to it have no effect.

8.9.4. USB BRIDGE PCI STATUS REGISTER

This is the 16-bit PCI Status register.

USBB_Stat Access = 0xCF8h/0xCFCh Regoffset = 0x6h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DP	SS	SMA	RTA	STA	DT		DPED	FBBC	Rsv						
Default value after reset = 0280h															

Bit Number	Mnemonic	Description
Bit 15	**DP**	**Detected parity error.** This bit is set when a PCI parity error is detected. It may be cleared by software by writing a ‘1’ back to this bit.
Bit 14	**SS**	**Signalled SERR#.** This bit is set to a ‘1’ when SERR# is asserted by the USB Bridge. Writing a ‘1’ to this bit will clear it.
Bit 13	**SMA**	**Signalled Master Abort.** This bit is set to a 1 when the USB Bridge terminates a PCI transaction with a master abort. Writing a ‘1’ to this bit will clear it.
Bit 12	**RTA**	**Received Target Abort.** This bit is set to a ‘1’ when PCI transaction initiated by the USB Bridge is terminated with a target abort. Writing a ‘1’ to this bit will clear it.
Bit 11	**STA**	**Signalled Target Abort.** This bit is hardwired to ‘0’.
Bits 10-9	**DT**	**DEVSEL Timing.** These bits are hardwired for medium timing to ‘01’. Writes have no effect.
Bit 8	**DPED**	**Data Parity Error Detected.** This bit is set to ‘1’ when a PCI data parity error is detected. Writing a ‘1’ will clear it.
Bit 7	**FBBC**	**Fast Back-to-Back Capable.** Hardwired to ‘1’. Indicates that the USB Bridge, while acting as target, is capable of accepting fast back-to-back transactions. Reads will always return ‘1’, writes have no effect.
Bits 6-0	**Rsv**	**Reserved.** These bits are hardwired to ‘0’s.

8.9.5. USB BRIDGE PCI REVISION ID REGISTER

This is the 8-bit read only PCI revision identification register.

USBB_R_ID Access = 0xCF8h/0xCFCh Regoffset = 0x8h

7	6	5	4	3	2	1	0
Rsv							
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-0	**Rsv**	**Reserved.** These bits are hardwired to 00h.

8.9.6. USB BRIDGE DEVICE CLASS CODE REGISTER

This is a 24-bit read only register implemented at configuration space offset 9h, Ah, Bh.

USBB_C_Code Access = 0xCF8h/0xCFCh Regoffset = 0x9h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BCC								SCC							
Default value after reset = 0Ch								Default value after reset = 03h							

15	14	13	12	11	10	9	8
PIR							
Default value after reset = 10h							

Bit Number	Mnemonic	Description
Bits 31-24	**BCC**	**Base Class Code.** These bits are hardwired to 0Ch.
Bits 23-16	**SCC**	**Sub Class Code.** These bits are hardwired to 03h.
Bits 15-8	**PIR**	**Programming Interface Register.** These bits are hardwired to 10h.

8.9.7. USB BRIDGE HEADER TYPE REGISTER

This is an 8-bit read only register, hardwired to 40h

.

USBB_Head Access = 0xCF8h/0xCFCh Regoffset = 0xEh

7	6	5	4	3	2	1	0
0	1	0	0	0	0	0	0

8.10. PCI Configuration for OpenHCI-compliant USB Host Controller

The following table provides a summary of the registers that are necessary for the USB Host Controller to be successfully configured in a PCI-based PC host. Those registers which are implementation-dependent are not described in the table; their implementation is left to the individual manufacturers for innovation. However, they are defined in the PCI Specification, Revision 2.1.

Register	Offset	Description
Command	**05 - 04**	Provides coarse control over a device’s ability to generate and respond to PCI cycles
CLASS_CODE	**0B - 09**	Identifies the generic function of the device
BAR_OHCI	**13 - 10**	Specifies the base address of a contiguous block in the main memory of the PC host, from which 4 KB of directly-mapped addressing spaces are reserved by OpenHCI for the operational registers of the Host Controller

8.10.1. COMMAND REGISTER

This register provides coarse control over the ability to generate and respond to PCI cycles. It is imperative that the Host Controller supports both PCI bus-mastering and memory-mapping of all operational registers into the main memory of the PC host. Consequently, the fields **MA** and **BM** should always be set to ‘1’ during device configuration.

Once the Host Controller has started processing endpoint lists of periodic and nonperiodic, the action to reset either field **MA** or **BM** to its default value should be approached with caution. If the field **MA** is reset to ‘0’, the Host Controller can no longer respond to any software command addressed to it and interrupt generation is halted, while the Host Controller can still generate the SOF token at the beginning of each frame. If the field **BM** is reset to ‘0’, the Host Controller will no longer be able to read Distributors (both Endpoint and Transfer) from the main memory, nor can it update the *HCCA* partition in the main memory.

Command Access = h Regoffset = 0xh

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
*							Rsv						BM	MA	*
Default value after reset = 0															

Bit Number	Mnemonic	Description
Bits 31 - 16	**Rsv**	Reserved
Bits 15 - 9	*****	Refer to PCI Specification, Revision 2.1, for definition
Bits 8 - 3	**Rsv**	Reserved
Bit 2	**BM**	**BUS MASTER** **Default ‘0’ Indicates the device’s ability to act as a bus-master**
Bit 1	**MA**	MEMORY ACCESS **Default ‘0’** Indicates the device’s ability to respond to PCI memory cycles
Bit 0	*****	Refer to PCI Specification, Revision 2.1, for definition

8.10.2. CLASS_CODE Register

This register identifies the basic function of the device, and a specific programming interface code for a USB Host Controller.

CLASS_CODE Access = h Regoffset = 0xh

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv								BC							
Default value after reset = 0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SC								PI							
Default value after reset = 0															

Bit Number	Mnemonic	Description
Bits 31 - 24	**Rsv**	Reserved
Bits 23 - 16	**BC**	**BASE CLASS** A constant value of ‘0Ch’ Identifies the device being a Serial Bus Controller
Bits 15 - 8	**SC**	**SUB CLASS** A constant value of ‘03h’ Identifies the device being of Universal Serial Bus
Bits 7 - 0	**PI**	**PROGRAMMING INTERFACE** A constant value of ‘10h’ Identifies the device being a Host Controller

8.10.3. BAR_OHCI Register

The *BAR_OHCI* register specifies the base address of a contiguous memory space in the main memory of the PC host, which is reserved for the operational registers defined by the Specification, Release 1.0. All of the operational registers are directly mapped into this memory space. With reference to the PCI Specification, Revision 2.1, the Host Controller Driver will always allocate a memory band of 4 KB for the operational registers. This is despite the fact that the number of operational registers defined by the Specification, Release 1.0, is far less than 4 KB. Regardless of whether the hardware vendor of a USB Host Controller chooses to implement the decoding logic for bits [11:0] or not, the respective hardware **must** be able to decode the operational registers. When any of the addresses between the block of operational registers and the 4-KB upper-bound is accessed, the hardware is not required to respond and the access can be ignored.

The hardware registers that are used to implement vendor specific features are not covered by the Specification, Release 1.0. Consequently, any vendor-specific hardware registers **should not** be mapped into the memory space starting at the address location as indicated by BAR_OHCI.

BAR_OHCI Access = h Regoffset = 0xh

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BAR															
Default value after reset = 0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BAR												PM	TP		IND
Default value after reset = 0															

Bit Number	Mnemonic	Description
Bits 31 - 12	**BAR**	**BASE ADDRESS** Specifies the upper 20 bits of the 32-bit starting base address. This represents a maximum of 4-KB addressing space for the OpenHCI's operational registers
Bits 11 - 4		Default value of '00h' and is read only Represents a maximum of 4-KB addressing space for the OpenHCI's operational registers
Bit 3	**PM**	**PREFETCH MEMORY** A constant value of '0' Indicates that there is no support for "prefetchable memory"

Bit Number	Mnemonic	Description
Bits 2 - 1	**TP**	**TYPE** A constant value of ‘00b’ Indicates that the base register is 32-bit wide and can be placed anywhere in the 32-bit memory space; i.e., lower 4 GB of the main memory of the PC host
Bit 0	**IND**	**INDICATOR** A constant value of ‘0b’ Indicates that the operational registers of the device are mapped into memory space of the main memory of the PC host system

8.11. Legacy USB Support Registers

Four operational registers are used to provide USB legacy support. Each of these registers is located on a 32-bit boundary. The offset of these registers is relative to the base address of the Host Controller operational registers with *HceControl* located at offset 100h.

Register	Offset	Description
HceControl	100h	Used to enable and control the emulation hardware and report various status information.
HceInput	104h	Emulation side of the legacy Input Buffer register.
HceOutput	108h	Emulation side of the legacy Output Buffer register where keyboard and mouse data is to be written by software.
Hce Status	10Ch	Emulation side of the legacy Status register.

9. ISA INTERFACE

9.1. INTRODUCTION

The ISA Interface provides access to the peripherals available in the STPC device and to Memory and external devices on the ISA bus.

Control of the ISA bus is by the North Bridge which acts as a bridge between the host CPU bus and the PCI bus. Reads and writes which are initiated by the CPU are subtractively decoded. Reads and writes that target North Bridge internal registers or main memory are routed to those targets, and all other reads and writes are sent to the PCI bus. The cycles for interrupt acknowledge, shutdown, stop grant and halt are also sent to the PCI bus.

The North Bridge also routes PCI reads and writes to cache, main memory and its internal registers. The South Bridge acts as a bridge between the PCI bus and the ISA bus. ISA bus cycles may be initiated by PCI bus cycles, or by an ISA bus card. Additionally, refresh cycles are run periodically by the ISA controller.

The South Bridge will claim all PCI cycles which were initiated outside the South Bridge and not claimed by any other PCI slave. Reads and writes to PCI configuration registers are routed appropriately by the South Bridge's PCI controller. All other PCI operations, including reads and writes to the South Bridge internal registers, are sent to the ISA controller. With the exception of writes to the keyboard controller under certain conditions, a read or write cycle sent to the ISA bus controller will create one or more ISA bus cycles.

Because of the speed difference between ISA bus and PCI bus, and the requirement that PCI cycles be less than a certain number of clocks, PCI cycles which go to the ISA bus will require retries on the PCI bus.

The cycles for interrupt acknowledge, shutdown, stop grant and halt are also sent to the ISA bus controller. These cycles do not create ISA bus cycles, but they use the same state machines for timing and arbitration as reads and writes. ISA bus cycles which are initiated by an ISA bus card are either DMA cycles, in which case the address is supplied by the DMA controller, or ISA bus master cycles, in which case the address is supplied by the card itself.

Every cycle initiated on the ISA bus is tried on the PCI bus. If the cycle is claimed by some PCI target, then data is read from or written to that target. If the PCI cycle is not claimed, and the cycle targets a South Bridge internal register, then that register is read from or written to. Otherwise, the target is expected to be on the ISA bus.

9.2. PCI / ISA CYCLES

9.2.1. PCI TO ISA READ AND WRITE

The PCI transfers data four bytes at a time, with byte enables for each byte. The South Bridge's PCI controller transfers these four bytes and four byte enables to the ISA controller. The ISA controller in turn runs zero to four ISA cycles. For 8-bit targets, the enabled bytes are read or written in order, least significant byte (lowest address) first.

For 16-bit targets, enabled bytes are again read or written in order, but a 16-bit transfer is used when an even byte is enabled and the following odd byte is also enabled.

Eight-bit ISA operations are by default four and a half ISACLK cycles, starting on a falling edge of ISACLK and ending on a rising edge. Sixteen-bit cycles are by default two and a half ISACLK cycles, also starting on a falling edge of ISACLK and ending on a rising clock. An additional clock cycle may be added by setting bit 5 in Index Register 50. Cycles can also be extended by pulling IOCHRDY low.

9.2.2. PCI TO INTERNAL REGISTER READ AND WRITE

All South Bridge internal registers are 8-bit. If an IO read or write targets an internal register, the target is assumed to be 8-bit wide (that is IOCS16# is ignored). Timing for reads and writes to internal registers is the same as 8-bit cycles on the ISA bus (see Section 9.2.1.).

If a write targets an internal register of the South Bridge, the data is written to the register and also to the ISA bus. If a read targets an internal register, the internal register is read, the South Bridge drives the ISA data bus with the contents of the register, and a ISA read cycle is done.

Registers that are called index registers in this document are indirectly addressed through a register at IO address 22h. There are two copies of this register, one on the North Bridge and one on the South Bridge.

Writes to IO address 22h go to both copies of the register. Reads from IO address 22h normally come from the North Bridge copy of the register, and do not generate a read cycle on the PCI bus. For test purposes, this behaviour can be changed by setting bit zero of index register 21h. In this case, a read from IO address 22h reads the South Bridge copy of the register, using a PCI read cycle.

After selecting an index register by writing to IO address 22h, that index register is read from or written to at IO address 23h. Some index registers are implemented in the North Bridge alone, some in the South Bridge alone, and some are duplicated and implemented in both. Whether an index register is implemented in the North Bridge, South Bridge or both is indicated in the description of that register in this document.

For index registers that are implemented in the North Bridge alone, writes to IO address 23h write to the register, and reads of IO address 23h read from the register, and no PCI cycles are generated.

For index registers that are implemented in the South Bridge alone, writes to IO address 23h write to the register, and reads of IO address 23h read from the register. In both cases, the data must go over the PCI bus.

For index registers that are implemented in both the North Bridge and the South Bridge, writes to IO address 23h write to both copies of the register, requiring a PCI write cycle. Reads to IO address 23h reads from the North Bridge copy of the register, and generate no PCI cycles. For test purposes, this behaviour can be changed by setting bit zero of index register 21h. In this case, the South Bridge copy of the register is read, using a PCI read cycle.

9.2.3. INTERRUPT ACKNOWLEDGE CYCLE

When an interrupt is requested, the interrupt controller in the South Bridge asserts the CPU's interrupt input. When the CPU services the interrupt, it must first get the interrupt vector from the interrupt controller. The interrupt vector is used to find the interrupt service routine. Also, since each interrupt request input of the interrupt controller has its own interrupt vector, the vector tells where the interrupt request came from.

To get the interrupt vector, the CPU generates two interrupt acknowledge cycles. Both of these cycles read data from the interrupt controller. The data returned by the first is ignored, while the data for the second contains the interrupt vector in bits 0-7. The North Bridge handles both of the cycles identically, converting them to PCI interrupt acknowledge cycles.

Outside of the interrupt controller, the South Bridge handles both cycles identically. The ISA controller converts the PCI cycles into interrupt acknowledge cycles for the interrupt controllers. The INTA# input of the interrupt controller is asserted for four and a half ISA bus clocks, starting on a falling edge of that clock, and during this time data is transferred from the interrupt controller to the ISA controller. This can be extended to five and a half clocks by setting bit 5 in Index Register 50h.

9.2.4. ISA TO PCI READ AND WRITE

ISA initiated cycles are converted to PCI cycles by the ISA controller. The South Bridge pulls IOCHRDY low to extend these cycles until the PCI cycle has completed.

9.2.5. ISA TO PCI BUFFERED READS

ISA reads of host memory can be buffered. This is disabled by default, and can be enabled by setting bit 6 in Index Register 50h. When this bit is set, ISA bus initiated reads of host memory addresses always get their data from a four byte buffer in the ISA controller which is filled on demand. This can reduce the amount of traffic for a block memory read by up to a factor of four.

The buffer is filled or refilled, under the conditions listed below, after the start of a ISA initiated read of a host memory address has been detected by the South Bridge. The South Bridge generates a PCI read of four bytes, with the low two bits of the address set to zero, and the rest of the address set to be the same as the address on the ISA bus address. The requested data will be driven by the South Bridge onto the ISA bus to finish the ISA read cycle.

The buffer will be refilled if the data requested by the current read is not in the buffer. Also, to avoid stale data, the buffer will be refilled for:

- The first host memory read after an ISA bus master gets ownership of the bus,
- The first host memory read after any ISA bus cycle which is not a host memory read,
- Any ISA read of a byte in the buffer which has already been read since the buffer was last filled.

If a host memory read can be fulfilled without refilling the buffer, no PCI cycle is generated.

9.2.6. ISA TO PCI POSTED WRITES

ISA writes to host memory can be posted. This is disabled by default, and can be enabled by setting bit 7 in Index Register 50h. When this bit is set, ISA bus initiated writes to host memory addresses go to a four byte buffer in the ISA controller. No PCI write is generated until the buffer is written to host memory.

The buffer is written to host memory when:

- The buffer gets full,
- or there is a host memory write to a location not in the buffer,
- or a host memory write would overwrite data already in the buffer,
- or there is an ISA cycle which is not a host memory write,
- or the current ISA master gives up ownership of the bus.

If writing the buffer to host memory is triggered by an ISA bus cycle, that cycle is held up by pulling IOCHRDY low until the buffer has been written to host memory.

Note that it is possible for the South Bridge to generate writes with discontiguous byte enables if posted writes are enabled.

9.2.7. ISA TO REGISTER READ AND WRITE

ISA initiated cycles which target South Bridge internal registers will first be tried on the PCI bus. If they are not claimed by a PCI target, then the register will be read or written. Reads and writes to IPC registers will cause the South Bridge to pull IOCHRDY low for at least the number of cycles programmed into Index Register 01h. Reads and writes to the South Bridge registers which are not IPC registers are normally

disabled. These can be enabled by setting bit 7 of Index Register 51h. Writes to these registers require a longer than standard recovery time of two ISACLK periods.

9.3. XBUS READ AND WRITE

The XBUS is an 8-bit subset of the ISA bus that connects low speed devices on the mother board to the CPU. In particular, the Real Time Clock (RTC), the Keyboard Controller, and the BIOS ROM will usually be connected via the XBUS. For the STPC, the XBUS shares address, data and command lines with the ISA bus. No buffers or transceivers are required to connect the XBUS to the ISA bus. The timing for XBUS cycles is the same as that for eight bit ISA cycles, see above.

9.3.1. REAL TIME CLOCK READ AND WRITE

The Real Time Clock (RTC) is connected to the XBUS. However the RTC is not connected to the command lines of the XBUS. Instead, four input pins of the RTC (CS#, AS, RW#, DS) are controlled directly by the STPC. The MOT pin of the RTC must be tied low. The registers in the RTC are accessed indirectly, by first writing the register number to IO port 70h, and then reading or writing the register at IO port 71h.

The RTC input CS# is connected to the logical OR of the outputs RMRTCCS# and ISAOE#. CS# is the chip select for the RTC, and it will be driven low (active) on any IO read or write to port 70h or port 71h, and also will be driven low by reads or writes to ROM address space.

The RTC input AS is directly connected to the RTCAS output. AS is the address strobe for the RTC, and it is asserted (high) during any IO write to port 70h.

The RTC input RW# is connected to the logical OR of the RTCRW# and ISAOE# outputs. RW# is write pulse for the RTC, and it will be asserted (low) during any IO write to port 71h.

The RTC input DS is connected to the logical OR of the South Bridge outputs RTCDS and ISAOE#. DS is the read pulse for the RTC, and it will be asserted (low) during any IO read of port 71h.

The RTC interrupt output IRQ# is directly connected to the IRQ8 input. There is an internal inverter between the pin IRQ8 and the interrupt controller to maintain compatibility with the PC-AT without requiring additional external glue logic.

9.3.2. BIOS ROM READ AND WRITE

The BIOS ROM is connected to the XBUS. The chip select for the ROM is connected to the logical OR of the RMRTCCS# and ISAOE# outputs.

9.3.3. CPU RESET AND GATE A20

Before the 286 CPU, memory space was limited to 1MB. Some software applications used this characteristic to access data in segment 0 by generating an address above the 1MB. To stay compatible with these applications, when the 286 appeared, the PC motherboards included, via the keyboard controller, a mechanism to enable or disable this pre-286 compatibility. This is done by the 'Gate A20' mechanism. When enabled, the CPU A20 address line is propagated to the memory bus. When disabled, the memory bus A20 line is forced to 0 (8086 compatible).

To be able to reset the CPU, the keyboard controller also includes a pin which is connected to the CPU reset pin (only the CPU is reset, not the chipset or external components).

The STPC doesn't provide external pins to be able to control the gate A20 and CPU reset. These features are controlled internally by the keyboard emulation. Thus, the STPC checks the commands and data sent to the keyboard controller, and when it recognizes the related commands, applies them internally. The keyboard emulation must be on (register STPC_MISC0/bit 3 = 0), else you won't be able to reset the CPU and the A20 line will always be masked. This is done in the default configuration.

Notes; Only the P2 write command (D1h) and the reset pulse command (FEh) are emulated. In particular, the P2 read command (D0h) is not emulated, so the return value is the keyboard controller P2 state.

On the other STPCs, the commands and data are forwarded, so the keyboard controller receives and applies them, the resulting actions are ignored by the STPC, but the keyboard controller A20 state reflects the STPC A20 state.

The gate A20 and CPU reset commands are not forwarded to the internal keyboard controller.

9.3.3.1. Reset Method

To disable Gate A20 (forcing address bit 20 to low), write D1h to the I/O Port 64h then write xxxx xx0xb to I/O Port 60h.
To enable Gate A20 (forcing address bit 20h to high

The Reset, also known as warm reset, is generated by writing data FEh to I/O port 64h or by writing data DEh to I/O port 64h then writing data xxxxxxx0 binary (bit 0 = ‘0’) to I/O port 60h.

Fast host CPU reset only is generated by two methods:

1) Whenever the STPC detects a write to Port 64h with data FEh.

2) Whenever the STPC detects a write to Port 60h following a D1h data write to Port 64h, bit 0 of the data byte being written at Port 60h is ‘0’.

The CPU reset is at least 16 host clocks. The write cycle is not forwarded to the keyboard controller.

9.4. ISA STANDARD REGISTERS

The ISA standard registers correspond to the registers in the peripheral components integrated in the STPC as well as the miscellaneous ports implemented on a ISA motherboard. These registers reside in IO space.

The functions controlled by the ISA registers include the DMA and interrupt control, BIOS and keyboard interface.

9.4.1. DMA 1 CONTROLLER REGISTERS

DMA 1 controls 8-bit DMA transfers.

There are 16 DMA 1 registers. They are as shown in Table 9-1.

Table 9-1. DMA1 Registers

IO address bits 15-0	Reset Value	Register Name	Mnemonic
XXXX XX00 000x 0000	xxxx xxxx	DMA 1 Channel 0 Base and Current Address	DMA1_CBA0
XXXX XX00 000x 0001	xxxx xxxx	DMA 1 Channel 0 Base and Current Count	DMA1_CBC0
XXXX XX00 000x 0010	xxxx xxxx	DMA 1 Channel 1 Base and Current Address	DMA1_CBA1
XXXX XX00 000x 0011	xxxx xxxx	DMA 1 Channel 1 Base and Current Count	DMA1_CBC1
XXXX XX00 000x 0100	xxxx xxxx	DMA 1 Channel 2 Base and Current Address	DMA1_CBA2
XXXX XX00 000x 0101	xxxx xxxx	DMA 1 Channel 2 Base and Current Count	DMA1_CBC2
XXXX XX00 000x 0110	xxxx xxxx	DMA 1 Channel 3 Base and Current Address	DMA1_CBA3
XXXX XX00 000x 0111	xxxx xxxx	DMA 1 Channel 3 Base and Current Count	DMA1_CBC3
XXXX XX00 000x 1000	xxxx 0000	DMA 1 Read Status/Write Command register	DMA1_RSWC
XXXX XX00 000x 1001	1111 xxxx	DMA 1 Request register	DMA1_RR
XXXX XX00 000x 1010	0000 0000	DMA 1 Read Command/Write Single Mask register	DMA1_RCWSM
XXXX XX00 000x 1011	0000 0000	DMA 1 Mode register	DMA1_Mode
XXXX XX00 000x 1100	1111 1111	DMA 1 Set/Clear Byte pointer flip-flop	DMA1_SCBPFF
XXXX XX00 000x 1101	0000 0000	DMA 1 Read Temp register/Master Clear	DMA1_RTMC
XXXX XX00 000x 1110	1111 1111	DMA 1 Clear Mask/Clear all request	DMA1_CMCAR
XXXX XX00 000x 1111	1111 1111	DMA 1 Read/Write all Mask register bits	DMA1_RWMB

Note that the not all bits of the address are used.

Programming notes:

Channel 0 corresponds to the internal DRQ0B, channel 1 to DRQ1B, channel 2 to DRQ2B, and channel 3 corresponds to the internal DRQ3B.

9.4.2. INTERRUPT CONTROLLER 1 REGISTERS

There are two interrupt controller 1 registers. They are as shown in Table 9-2.

Interrupt controller 1 is the master interrupt controller.

Table 9-2. Interrupt Controller 1 Registers

IO address bits 15-0	Reset Value	Register Name	Mnemonic
0000 0000 0010 0000	0000 0000	Interrupt Controller 1 register	IC_1
0000 0000 0010 0001	1111 1111	Interrupt Controller 1 Mask register	IC_1MR

Note that not all bits of the address are used.

Programming notes:

Interrupt controller 1 input IR0 is connected IRQ0, IR1 to IRQ1, IR2 to interrupt out from interrupt controller 2, IR3 to IRQ3, IR4 to IRQ4, IR5 to IRQ5, IR6 to IRQ6, and IR7 to IRQ7.

9.4.3. INTERVAL TIMER REGISTERS

The Interval Timer comprises three independent counters. Counter 0 is used to generate timer interrupts, counter 1 is used to generate ISA bus refresh, and counter 2 is used to create the speaker tone.

There are 4 Interval Timer registers. They are as shown in Table 9-3.

Table 9-3. Interval Timer Registers

IO address bits 15-0	Reset Value	Register Name	Mnemonic
XXXX XX00 010x xx00	xxxx xxxx	Interval Timer Register Counter 0 Count	IT_0
XXXX XX00 010x xx01	xxxx xxxx	Interval Timer Register Counter 1 Count	IT_1
XXXX XX00 010x xx10	xxxx xxxx	Interval Timer Register Counter 2 Count	IT_2
XXXX XX00 010x xx11	1111 1111	Command Mode register	IT_3

Note that not all bits of the address are decoded.

Programming notes:

All three counters are clocked by 1.193 MHz nominal frequency (OSC/12). Counter 0 and counter 1 gates are always on, counter 2 gate is controlled by writing to Port B (see Section 9.4.4.).

9.4.4. PORT B REGISTER

This is the ISA compatible 8-bit Port B register located at xxxx xxxx 0110 xxx1 IO address (bits 15-0). It has the following meaning:

Port_B Access = 0061h Regoffset =

7	6	5	4	3	2	1	0
PE	IOCHK	T/C 2S	ISA RC	ISA IOCHK	PCE	SE	T/C 2 G
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	PE	**Parity Error.** This bit is set to a ‘1’ whenever a parity error is detected during system memory read operation. Once set, this bit can be cleared by setting bit 2 of this register to a ‘1’. Bit 2 should be reset to a ‘0’ to enable recording the next parity error. The parity error generates NMI to the host CPU if NMI is enabled.This bit is read-only.
Bit 6	IOCHK	ISA IOCHK# Enable. This bit is set to a ‘1’ when IOCHK# signal of the ISA bus is asserted. Once set, this bit is cleared by setting bit 3 of this register to a ‘1’. Bit 3 should be reset to a ‘0’ to enable recording the next IOCHK#. IOCHK# generates NMI to the host CPU if NMI is enabled. This bit is read only.
Bit 5	T/C 2S	ISA T/C 2 State. This bit reflects the output of Timer/Counter 2 without any synchronization. This bit is read only.
Bit 4	ISA RC	ISA Refresh Check. This bit toggles on every rising edge of the REFRESH# signal of the ISA bus. This bit is read only.
Bit 3	ISA IOCHK	ISA IOCHK# Enable. This bit is connected to the asynchronous clear input of the flipflop which records the IOCHK#. It must be set to a ‘1’ to clear the flipflop and then set to a ‘0’ to enable further IOCHK# assertions. This bit is read/write and cleared to a ‘0’ by ISA reset.
Bit 2	PCE	**Parity Check Enable.** This bit is connected to the asynchronous clear input of the flipflop which records the parity error. It must be set to a ‘1’ to clear the flipflop and then set to a ‘0’ to enable further parity errors. This bit is read/write and cleared to a ‘0’ by ISA reset.
Bit 1	SE	ISA Speaker Enable. This bit is ANDed with the Interval Timer counter 2 OUT signal to drive the Speaker output signal. This bit is read/write and cleared to a ‘0’ by ISA reset.
Bit 0	T/C 2G	T/C 2 Gate. This bit is connected to the gate input of the Interval Timer counter 2. This bit is read/write and cleared to a ‘0’ by ISA reset.

9.4.5. PORT 70H REGISTER

This 8-bit write-only register contains the NMI enable bit and is located at xxxx xxxx 0111 0xx1 IO address.

Port_70 Access = 0070h Regoffset =

7	6	5	4	3	2	1	0
NMI E	Rsv						
Default value after reset = 80h							

Bit Number	Mnemonic	Description
Bit 7	NMI E	NMI Enable. NMI is asserted on encountering IOCHK# on the ISA bus (Port_ B) or SERR# on the PCI bus if this bit is set to a ‘0’. Setting this bit to a ‘1’ disables NMI generation.
Bit 6-0	Rsv	**Reserved.** must be written to ‘0’s. Read back is undefined.

Programming notes:

Writing to this address also sets the address register in the Real Time Clock (RTC, not part of the STPC, normally connected via the ISA interface).

9.4.6. INTERRUPT CONTROLLER 2 REGISTERS

Interrupt controller 2 is the slave interrupt controller.

Interrupt controller 2 occupies two register locations. They are as shown in Table 9-4.

Table 9-4. Interrupt Controller 2 Registers

IO address bits 15-0	Reset Value	Register Name	Mnemonic
XXXX XX00 101x xxx0	0000 0000	Interrupt Controller 2 register	IC_2R
XXXX XX00 101x xxx1	1111 1111	Interrupt Controller 2 Mask register	IC_2M

Note that not all address bits are decoded.

Programming notes:

Interrupt controller 2 input IR1 is connected to IRQ9, IR2 to IRQ10, IR3 to IRQ11, IR4 to IRQ12, IR6 to IRQ14, IR7 to IRQ15. IR0 driven by IRQ8 inverted. IR5 is driven by an internally generated floating point error interrupt request.

9.4.7. DMA CONTROLLER 2 REGISTERS

There are 16 DMA 2 registers. They are as shown in Table 9-5.

Table 9-5. DMA Controller 2 Registers

IO address bits 15-0	Reset Value	Register Name	Mnemonic
XXXX XX00 1100 000x	xxxx xxxx	DMA 2 Channel 0 Base and Current Address	DMA2_CBAO
XXXX XX00 1100 001x	xxxx xxxx	DMA 2 Channel 0 Base and Current Count	DMA2_CBCO
XXXX XX00 1100 010x	xxxx xxxx	DMA 2 Channel 1 Base and Current Address	DMA2_CBA1
XXXX XX00 1100 011x	xxxx xxxx	DMA 2 Channel 1 Base and Current	DMA2_CBC1
XXXX XX00 1100 100x	xxxx xxxx	DMA 2 Channel 2 Base and Current Address	DMA2_CBA2
XXXX XX00 1100 101x	xxxx xxxx	DMA 2 Channel 2 Base and Current	DMA2_CBC2
XXXX XX00 1100 110x	xxxx xxxx	DMA 2 Channel 3 Base and Current Address	DMA2_CBA3
XXXX XX00 1100 111x	xxxx xxxx	DMA 2 Channel 3 Base and Current Count	DMA2_CBC3
XXXX XX00 1101 000x	1111 xxxx	DMA 2 Read Status/Write Command register	DMA2_RSWC
XXXX XX00 1101 001x	0000 0000	DMA 2 Request register	DMA2_RR
XXXX XX00 1101 010x	0000 0000	DMA 2 Read Command/Write Single Mask register	DMA2_RCWSM
XXXX XX00 1101 011x	0000 0000	DMA 2 Mode register	DMA2_Mode
XXXX XX00 1101 100x	1111 1111	DMA 2 Set/Clear Byte pointer flip-flop	DMA2_SCBPFF
XXXX XX00 1101 101x	0000 0000	DMA 2 Read Temporary/Master Clear	DMA2_RTMC
XXXX XX00 1101 110x	1111 1111	DMA 2 Clear Mask/Clear all requests register	DMA2_CMCAR
XXXX XX00 1101 111x	1111 1111	DMA 2 Read/Write all Mask register bits	DMA2_RWMRB

Note that the not all bits of the address are used.

9.4.8. DMA PAGE REGISTERS

The DMA Page registers defines address bits [16-23] for DMA transfers controlled by DMA 1 or DMA 2. Bits [0-15] are generated by the DMA controller, bits [16-23] come from the appropriate page register, and bits 31-24 are all zeroes.

There are 16 DMA page registers. They are as shown in Table 9-6.

Table 9-6. DMA Page Registers

IO address bits 15-0	Reset Value	Register Name	Mnemonic
XXXX XX00 1000 0000	xxxx xxxx	DMA Page Register Port 80h (Reserved)	Port_80
XXXX XX00 1000 0001	xxxx xxxx	DMA Page Register Channel 2	DMA_PRC2
XXXX XX00 1000 0010	xxxx xxxx	DMA Page Register Channel 3	DMA_PRC3
XXXX XX00 1000 0011	xxxx xxxx	DMA Page Register Channel 1	DMA_PRC1
XXXX XX00 1000 0100	xxxx xxxx	DMA Page Register Port 84h (Reserved)	Port_84
XXXX XX00 1000 0101	xxxx xxxx	DMA Page Register Port 85h (Reserved)	Port_85
XXXX XX00 1000 0110	xxxx xxxx	DMA Page Register Port 86h (Reserved)	Port_86
XXXX XX00 1000 0111	xxxx xxxx	DMA Page Register Channel 0	DMA_PRC0
XXXX XX00 1000 1000	xxxx xxxx	DMA Page Register Port 87h (Reserved)	Port_87
XXXX XX00 1000 1001	xxxx xxxx	DMA Page Register Channel 6	DMA_PRC6
XXXX XX00 1000 1010	xxxx xxxx	DMA Page Register Channel 7	DMA_PRC7
XXXX XX00 1000 1011	xxxx xxxx	DMA Page Register Channel 5	DMA_PRC5

Table 9-6. DMA Page Registers

XXXX XX00 1000 1100	xxxx xxxx	DMA Page Register Port 8Bh (Reserved)	Port_8B
XXXX XX00 1000 1101	xxxx xxxx	DMA Page Register Port 8Ch (Reserved)	Port_8C
XXXX XX00 1000 1110	xxxx xxxx	DMA Page Register Port 8Dh (Reserved)	Port_8D
XXXX XX00 1000 1111	xxxx xxxx	DMA Page Register Port 8Eh (Reserved)	Port_8E

9.5. ISA CONFIGURATION REGISTERS

These registers are addressed through the Address Configuration Index (CI) and Data registers.

9.5.1. MISCELLANEOUS CONTROL REGISTER 0

Misc_Cont0 Access = 0022h/0023h Regoffset = 050h

7	6	5	4	3	2	1	0
ISA WPE	ISA RBE	ISA WIC	ISA CFS	KRE	CPU D		
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	ISA WPE	**ISA Write Post Enable**. If ‘1’, posted writes to host memory by ISA DMA or ISA bus master are enabled.
Bit 6	ISA RBE	**ISA Read Buffer Enable.** If ‘1’, buffered reads of host memory by ISA DMA or ISA bus master are enabled.
Bit 5	ISA WIC	**ISA Wait Insert Control.** This bit controls if extra wait state is inserted for slower ISA devices. 0: No extra wait state for ISA cycle 1: One extra wait state for ISA cycle
Bit 4	ISA CFS	**ISA Clock Frequency Select.** This bit selects the ISA clock frequency. 0: ISA clock is 14.31818 MHz / 2 1: ISA clock is PCICLK /4
Bit 3	KRE	**Keyboard Reset Enable.** This bit if set to a ‘1’, keyboard emulation fast gate A20 and fast reset are disabled. The source of warm reset indication is from the keyboard controller and the CPU core will use the gate A20 indication from keyboard controller for its internal A20M# input.
Bits 2-0	CPU D	**CPU Deturbo.** These three bits define the ratio CPU is held. (see Table 9-7).

Table 9-7. CPU Deturbo

Bit 2	Bit 1	Bit 0	CPU Deturbo
0	0	0	deturbo is disabled.
0	0	1	CPU is held 1/2 of the time.
0	1	0	CPU is held 2/3 of the time.
0	1	1	CPU is held 3/4 of the time.
1	0	0	CPU is held 4/5 of the time.
1	0	1	CPU is held 5/6 of the time.
1	1	0	CPU is held 6/7 of the time.
1	1	1	CPU is held 7/8 of the time.

9.5.2. MISCELLANEOUS CONTROL REGISTER 1

Misc_Cont1 Access = 0022h/0023h Regoffset = 051h

7	6	5	4	3	2	1	0
IPC W	CLK 24	HCLK D	Rsv	ROM	S E S	S D S	S C S
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	IPC W	**IPC Write control.** This bit controls the ISA master writes to the IPC register 0: ISA master writes to IPC register disabled 1: ISA master writes to IPC register enabled
Bit 6	CLK 24	**CLK24 Disable.** This bit controls the output of CLK24. 0: CLK24 generated normally 1: Clock synthesiser for CLK24 is disable (CLK24 will not toggle)
Bit 5	HCLK D	**HCLK Disable.** This bit controls the generation of HCLK. 0: HCLK generated normally 1: Clock synthesiser for HCLK is disabled (HCLK will not toggle)
Bit 4	Rsv	**Reserved**.
Bit 3	ROM	**ROM Write Protect Enable.** This bit, if set to a ‘1’, disables write cycles to ROM BIOS on extended bus. If set to ‘0’, write to extended bus ROM BIOS is allowed. Note: This bit can not disable the write to shadowed BIOS in DRAM since after shadow is enabled, all writes to BIOS should be forwarded to extended bus.
Bit 2	S E S	**Segment E Share.** This bit controls if E0000h-EFFFFh segment shares the FLASH memory with F0000h-FFFFFh segment. 0: Sharing disabled 1: Sharing enabled
Bit 1	S D S	**Segment D Share.** This bit controls if D0000h-DFFFFh segment shares the FLASH memory with F0000h-FFFFFh segment. 0: Sharing disabled 1: Sharing enabled
Bit 0	S C S	**Segment C Share.** This bit controls if C0000h-CFFFFh segment shares the FLASH memory with F0000h-FFFFFh segment. 0: Sharing disabled 1: Sharing enabled

9.5.3. PIRQ ROUTING CONTROL REGISTER 0

This 8-bit register controls the routing of PCI Interrupt A# to one of the interrupt inputs of the 8259 as follows. It applies to interrupts A# to D#:

PIRQ Routing	Regoffset
PIRQ A	52h
PIRQ B	53h
PIRQ C	54h
PIRQ D	55h

PAR_Cont0 Access = 0022h/0023h Regoffset = Table 8-1.

7	6	5	4	3	2	1	0
RE	Rsv			RC A			
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**RE**	**Routing Enable.** If set to a ‘1’, this bit enables the routing of PCI interrupt, otherwise the PCI interrupt A# is unconnected.
Bits 6-4	**Rsv**	**Reserved**. Writes have no affect. Reads return undefined value.
Bits 3-0	**RC A**	**Routing Control.** These bits route the PCI interrupt A# (see Table 9-8)

Table 9-8. Routing Control Encoding

Bit 3	Bit 2	Bit 1	Bit 0	Interrupt A# Route	Note
0	0	0	1	Reserved	1
0	0	0	1	Reserved	1
0	0	1	0	Reserved	1
0	0	1	1	IRQ3	
0	1	0	0	IRQ4	
0	1	0	1	IRQ5	
0	1	1	0	IRQ6	
0	1	1	1	IRQ7	
1	0	0	0	Reserved.	1
1	0	0	1	IRQ9	
1	0	1	0	IRQ10	
1	0	1	1	IRQ11	
1	1	0	0	IRQ12	
1	1	0	1	Reserved.	1
1	1	1	0	IRQ14	
1	1	1	1	IRQ15	

Note 1: Interrupt can not be routed to this level.

9.5.4. INTERRUPT LEVEL CONTROL REGISTER 0

This 8-bit register allows interrupt requests to the lower 8259 to be either level or edge sensitive on an interrupt by interrupt basis overriding the global edge/level control bit of the 8259.

IRQ_Lev_C_0 Access = 0022h/0023h Regoffset = 056h

7	6	5	4	3	2	1	0
IRQ C					Rsv		
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-3	**IRQ C**	**IRQ Control IRQ[7-3].** If set to a ‘1’, the corresponding interrupt request is treated as a level input, otherwise it is treated as the edge sensitive input (compatible to ISA).
Bits 2-0	**Rsv**	**Reserved.** Writes have no affect. Reads return undefined value.

9.5.5. INTERRUPT LEVEL CONTROL REGISTER 1

This register allows interrupt requests to the upper 8259 to be either level or edge sensitive on an interrupt by interrupt basis overriding the global edge/level control bit of the 8259. This register has the following definition.

IRQ_Lev_C_1 Access = 0022h/0023h Regoffset = 057h

7	6	5	4	3	2	1	0
IRQ C		Rsv	IRQ C				IPC
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-6	**IRQ C**	**IRQ Control IRQ[15-14].** If set to a '1', the corresponding interrupt request is treated as a level input, otherwise it is treated as the edge sensitive input (compatible with ISA).
Bit 5	**Rsv**	**Reserved.** Writes have no affect and the reads return undefined value.
Bits 4-1	**IRQ C**	**IRQ Control IRQ[12-9].** If set to a '1', the corresponding interrupt request is treated as a level input, otherwise it is treated as the edge sensitive input (compatible to ISA).
Bit 0	**IPC**	This bit controls the ISA refresh cycle.Setting to 0 disables ISA refresh and setting to 1 enables ISA refresh. By setting this bit to 1 enables the toggling the ISA Port B refresh bit (see Section 9.4.4.).

9.5.6. IPC CONFIGURATION REGISTER

This 8-bit register controls the timing of the DMA controllers, and also the number of wait states for writes to registers in the IPC.

IPC_Conf Access = 0022h/0023h Regoffset = 001h

7	6	5	4	3	2	1	0
IPC WS		DMA		DMA		DMA M	DMA C
Default value after reset = C0h							

Bit Number	Mnemonic	Description
Bits 7-6	**IPC WS**	**IPC Wait States.** These bits specify the number of ISACLK wait states for read or write to IPC register1 (see Table 9-9).
Bits 5-4	**DMA**	**DMA 16-Bit Wait States.** These bits specify the number of wait states in 16-bit DMA cycles (see Table 9-10).
Bits 3-2	**DMA**	**DMA 8-Bit Wait States.** These bits specify the number of wait states in 8 bit DMA cycle (see Table 9-11).
Bit 1	**DMA M**	**DMA MEMR# Timing.** If this bit is set to ‘1’ the DMA controllers will assert MEMR# at the the same time as IOW#. If set to ‘0’ (default), MEMR# will be asserted one clock after IOW#.
Bit 0	**DMA C**	**DMA Clock Select.** If this bit is set to ‘0’ (default), the DMA controller clock will be ISACLK divided by two, otherwise the DMA controller clock will be ISACLK.

Table 9-9. IPC Wait States

Bit 7	Bit 6	IPC Wait States
0	0	1
0	1	2
1	0	3
1	1	4 (Default)

Table 9-10. DMA 16-bit Wait States

Bit 5	Bit 4	DMA 16-bit Wait States
0	0	1 (Default)
0	1	2
1	0	3
1	1	4

Table 9-11. DMA 8-bit Wait States

Bit 3	Bit 2	DMA 8-bit Wait States
0	0	1 (Default)
0	1	2
1	0	3
1	1	4

Programming notes:

To read or write to this register, write 01 to index register 22h, and then read or write from data register 23h.

9.5.7. VMI IRQ ROUTING CONTROL REGISTER

This 8-bit register controls the routing of VMI Interrupt to one of the interrupt inputs of the 8259 as follows:

VIR_Conf Access = 0022h/0023h Regoffset = 058h

7	6	5	4	3	2	1	0
VMIE	Rsv			VMIC			
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**VMIR**	**VMI Routing Enable.** If set to a ‘1’, this bit enables the routing of VMI interrupt, otherwise the VMI interrupt is unconnected.
Bits 6-4	**Rsv**	**Reserved.** Writes have no affect. Reads return undefined value.
Bits 3-0	**VMIC**	**VMI Routing Control.** These bits route the VMI interrupt (see Table 9-12).

Table 9-12. VMI Routing Control Encoding

Bit 3	Bit 2	Bit 1	Bit 0	VMI Interrupt Route	Note
0	0	0	1	Reserved	1
0	0	0	1	Reserved	1
0	0	1	0	Reserved	1
0	0	1	1	IRQ3	
0	1	0	0	IRQ4	
0	1	0	1	IRQ5	
0	1	1	0	IRQ6	
0	1	1	1	IRQ7	
1	0	0	0	Reserved.	1
1	0	0	1	IRQ9	
1	0	1	0	IRQ10	
1	0	1	1	IRQ11	
1	1	0	0	IRQ12	
1	1	0	1	Reserved.	1
1	1	1	0	IRQ14	
1	1	1	1	IRQ15	
Note 1: Interrupt can not be routed to this level.					

9.5.8. ISA I/O PORT SELECT AND SYNC. REGISTER

This 8-bit register controls whether or not the signals between the PCI logic and the ISA logic are passed through synchronization logic. This register also controls UART and COM Port selection.

ISA_Sync Access = 0022h/0023h Regoffset = 059h

7	6	5	4	3	2	1	0
UBAS		UBAS1		Rsv		Rsv	SE
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-6	UBAS	**UART0 Base Address Selection.** (see Table 9-13).
Bits 5-4	UBAS1	**UART1 Base Address Selection.** (see Table 9-13).
Bits 3-2	Rsv	**Reserved**.
Bit 1	Rsv	**Reserved.** Returns a value of ‘0’ when read.
Bit 0	SE	**Synchronisation Enable.** 0: Enabled 1: Disabled

Table 9-13. UART COM Port Selection

Bit 7 (UART0) Bit 5 (UART1)	Bit 6 (UART0) Bit 4 (UART1)	COM Port Selection
0	0	COM1 (3F8h)
0	1	COM2 (2F8h)
1	0	COM3 (3E8h)
1	1	COM4 (2E8h)

Table 9-14. l COM Port IRQ Selection

UART0		UART1	
COM1	IRQ4	COM2, COM3 or COM4	IRQ3
COM2	IRQ3	COM1, COM3 or COM4	IRQ4
COM3	IRQ4	COM1, COM2 or COM4	IRQ3
COM4	IRQ3	COM1, COM2 or COM3	IRQ4

Programming notes:

Both the UARTs cannot be configured to the same BASE Address (there is a hardware protection). If this is attempted the UART0 will automatically default to COM1 and UART1 will default to COM2.

The bits (7-6-5-4) are used to generate chip select UARTs.

If the internal UARTs are disabled then these chip selects also go out of the chip for external use.

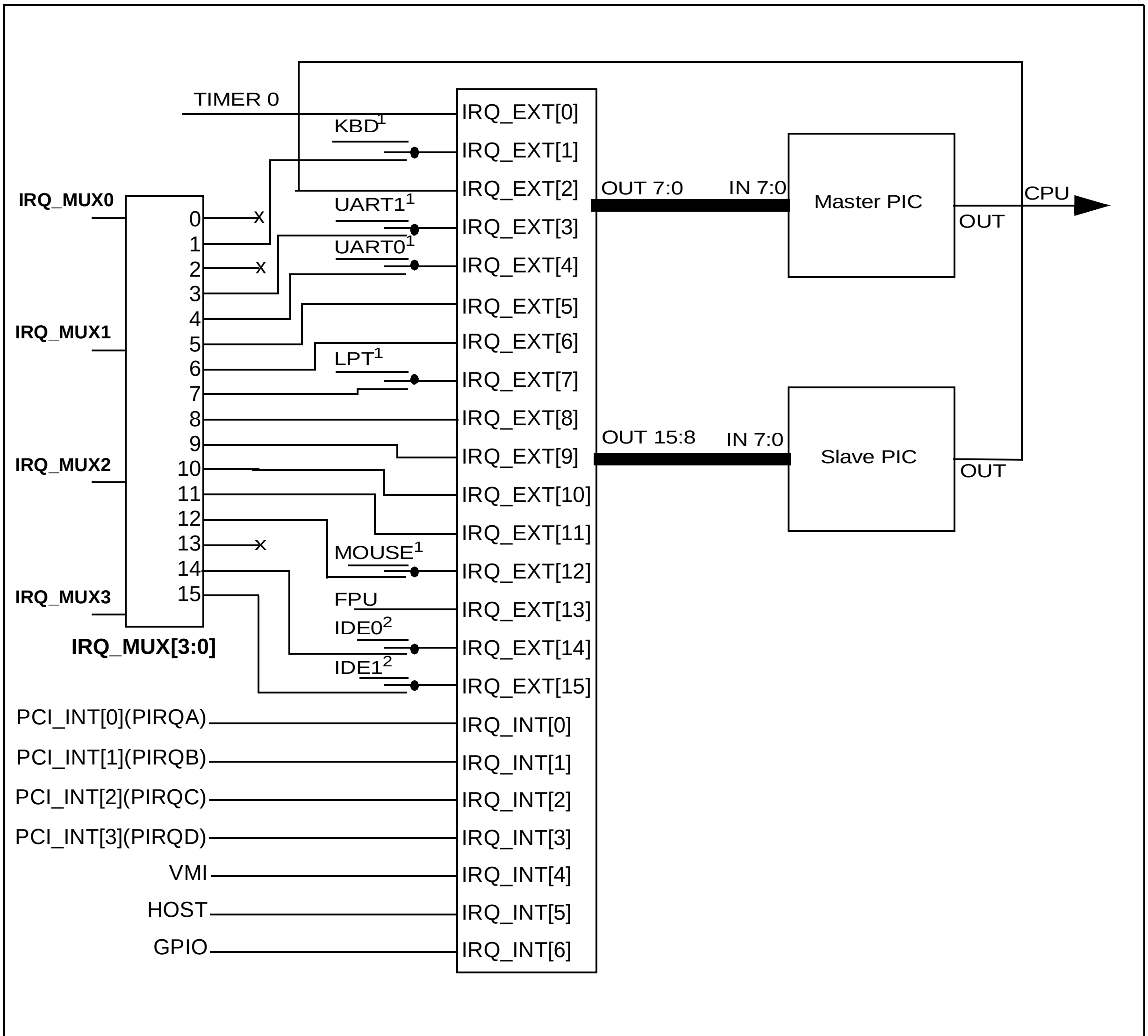
This bit would normally be set only when the ISA clock is derived from PCI clock (that is index 50h, bit 4 is set to ‘1’). Setting this bit will result in a small improvement in ISA performance.

9.6. INTERRUPT ROUTER

A general purpose crossbar is implemented for the routing of internal and external IRQ interrupts to any of the 16 inputs of two 8259 Programmable Interrupt Controller (PIC) devices. This is represented in Figure 9-1. The main interrupt features are listed below:

- No hardwired interrupt requirement
- Complete IRQ programming flexibility
- Interfaces with up to 11 internal (IRQ_INT[10:0]) and 16 external (IRQ_EXT[15-0]) interrupt requesting devices

Figure 9-1. Interrupt Router Schematic Layout



Note 1: These Interrupt lines are user available only when the Internal function are disabled (by Strap Option only).
Note 2: These Interrupt lines are dynamically assigned by software.

9.6.1. INTERRUPT CONTROLLER REGISTER SUMMARY

The Internal and External Interrupt Registers are set out as per the table below. The interrupts are layed out on their priority basis. If there are two or more input lines routed to the same output, only the highest priorty input will be taken into account, the lower inputs will be ignored. In Table 9-15. the highest priority is placed first the lowest priority is last.

Table 9-15. Interrupt Routing Control Registers

Register Offset/ Access	Default Value after Reset	Interrupt	Register Name	Mnemonic
Offset 052h[3]	0x00h	IRQ_INT[0]	PIRQA Routing Control Register	*PAR_cont0*
Offset 053h[3]	0x00h	IRQ_INT[1]	PIRQB Routing Control Register	*PBR_cont0*
Offset 054h[3]	0x00h	IRQ_INT[2]	PIRQC Routing Control Register	*PCR_cont0*
Offset 055h[3]	0x00h	IRQ_INT[3]	PIRQD Routing Control Register	*PDR_cont0*
Offset 058h[3]	0x00h	IRQ_INT[4]	VMI IRQ Routing Control Register	*VIR_cont*
0x410h	0x00h	IRQ_INT[5]	Host IRQ Routing Control Register	*IRQ_INT5*
0x411h	0x00h	IRQ_INT[6]	GPIO IRQ Routing Control Register	*IRQ_INT6*
0x412h	0x00h	IRQ_INT[7]	Not Implemented	*IRQ_INT7*
0x413h	0x00h	IRQ_INT[8]	Not Implemented	*IRQ_INT8*
0x414h	0x00h	IRQ_INT[9]	Not Implemented	*IRQ_INT9*
0x415h	0x00h	IRQ_INT[10]	Not Implemented	*IRQ_INT10*
0x400h	0x80h	IRQ_EXT[0]	Timer 0 IRQ Routing Control Register	*IRQ_EXT0*
0x401h	0x81h	IRQ_EXT[1]	Keyboard or IRQ External Routing Control Register1[2]	*IRQ_EXT1*
0x402h	0x82h	IRQ_EXT[2]	Slave PIC IRQ Routing Control Register	*IRQ_EXT2*
0x403h	0x83h	IRQ_EXT[3]	UART1 or IRQ External Routing Control Register 3[2]	*IRQ_EXT3*
0x404h	0x84h	IRQ_EXT[4]	UART0 or IRQ External Routing Control Register 4[2]	*IRQ_EXT4*
0x405h	0x85h	IRQ_EXT[5]	IRQ External Routing Control Register 5	*IRQ_EXT5*
0x406h	0x86h	IRQ_EXT[6]	IRQ External Routing Control Register 6	*IRQ_EXT6*
0x407h	0x87h	IRQ_EXT[7]	LPT or IRQ External Routing Control Register 7[2]	*IRQ_EXT7*
0x408h	0x88h	IRQ_EXT[8]	IRQ External Routing Control Register 8	*IRQ_EXT8*
0x409h	0x89h	IRQ_EXT[9]	IRQ External Routing Control Register 9	*IRQ_EXT9*
0x40Ah	0x8Ah	IRQ_EXT[10]	IRQ External Routing Control Register 10	*IRQ_EXT10*
0x40Bh	0x8Bh	IRQ_EXT[11]	IRQ External Routing Control Register 11	*IRQ_EXT11*
0x40Ch	0x8Ch	IRQ_EXT[12]	Mouse or Extenal IRQ Routing Control Register12[2]	*IRQ_EXT12*
0x40Dh	0x8Dh	IRQ_EXT[13]	FPU IRQ Routing Control Register	*IRQ_EXT13*
0x40Eh	0x8Eh	IRQ_EXT[14]	PCI IDE0 or External IRQ Routing Control Register[1]	*IRQ_EXT14*
0x40Fh	0x8Fh	IRQ_EXT[15]	PCI IDE1 or External IRQ Routing Control Register[1]	*IRQ_EXT15*
The pre-assigned interrupts follow the PC standard. See Section 6.3.6. in the Datasheet for routing details. The interrupts can be re-used internally provided they are not required for specific PC operations				
Note 1; These interrupts are not available when the IDE is in legacy mode				
Note 2; These interrupts are available for the ISA Bus only if the corresponding integrated function is deactivated by Strap Option (see STRAP OPTION chapter of the Datasheet)				
Note 3; These registers are accessable through chip set access 22h/23h.				

The interrupt router is connected between the interrupt-requesting devices and the two 8259 PIC devices. It requires 27 registers, one for each input. Five of these registers, Internal [0] to Internal [4], are implemented as PCI_IRQA, PCI_IRQB, PCI_IRQC, PCI_IRQD and VMI_IRQ respectively (highest priority interrupts listed in Table 9-15), in I/O space 22h/23h. The remaining 22 registers are implemented in I/O space from 400h to 415h. Details of all 27 registers are given in the following sections.

9.6.2. INTERRUPT ROUTING CONTROL REGISTERS

These eleven 8-bit registers control the routing of internal interrupts IRQ_INT[10:0] and the sixteen external interrupts IRQ_EXT[15:0] to the interrupt inputs of the 8259 as in Table 9-15. and Figure 9-1.

Routing Control Register Access = see Table 9-15 Regoffset see Table 9-15

7	6	5	4	3	2	1	0
RE	Rsv			RC			
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	RE	**Routing Enable.** If set to a ‘1’, this bit enables the routing of the internal interrupt, otherwise the internal interrupt is unconnected.
Bits 6-4	Rsv	**Reserved**. Writes have no affect. Reads return undefined value.
Bits 3-0	RC	**Routing Control.** These bits route the internal interrupt (see Table 9-8)

Table 9-16. Interrupt Route

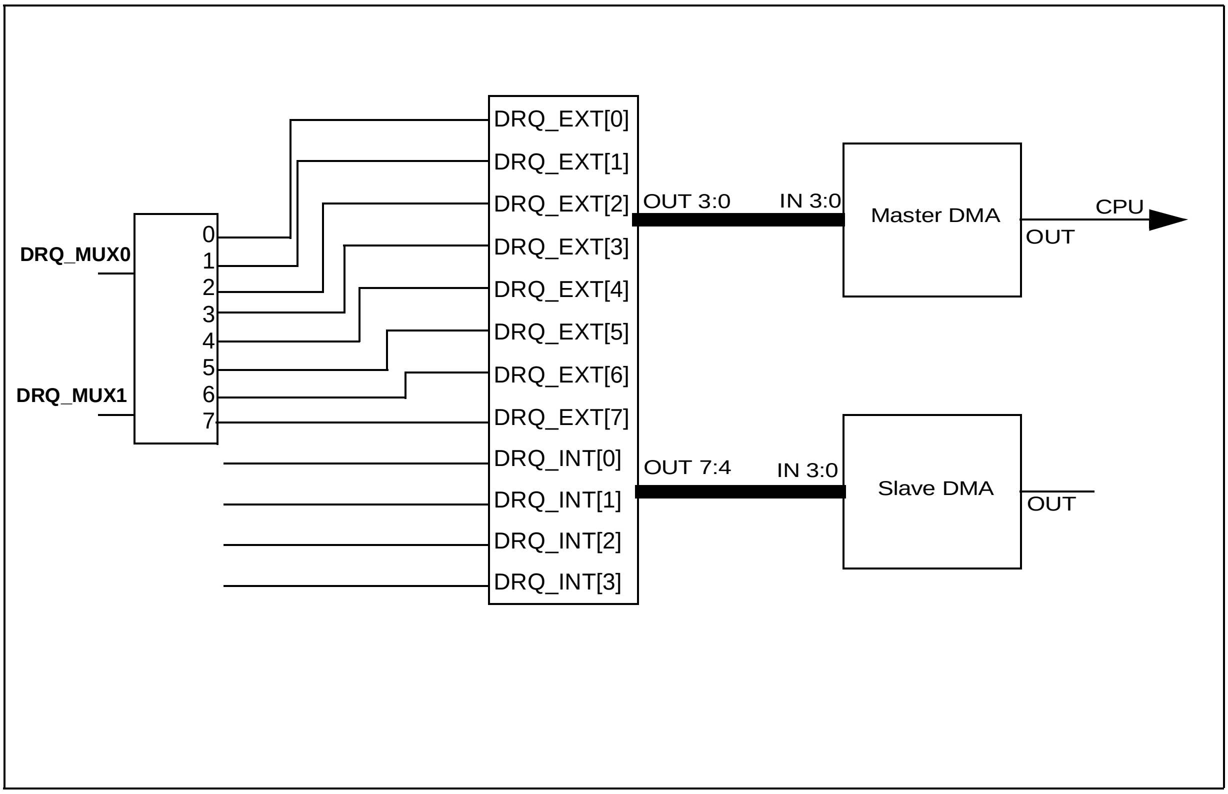
Bit 3	Bit 2	Bit 1	Bit 0	Interrupt Route
0	0	0	0	IRQ0
0	0	0	1	IRQ1
0	0	1	0	IRQ2
0	0	1	1	IRQ3
0	1	0	0	IRQ4
0	1	0	1	IRQ5
0	1	1	0	IRQ6
0	1	1	1	IRQ7
1	0	0	0	IRQ8
1	0	0	1	IRQ9
1	0	1	0	IRQ10
1	0	1	1	IRQ11
1	1	0	0	IRQ12
1	1	0	1	IRQ13
1	1	1	0	IRQ14
1	1	1	1	IRQ15

9.7. DRQ ROUTER

A general purpose crossbar is implemented for the routing of internal and external DRQs (DMA Requests) to any of the eight inputs of two 8237 DMA Controller devices. The main DRQ facility features are listed below:

- No hardwired DRQ requirement.
- Complete DRQ programming flexibility.
- Interfaces with internal and external DMA requesting devices.
- Number of DRQ lines expanded (in terms of the number of devices that can issue a DRQ) to eight external and four internal.

Figure 9-2. Interrupt Router Schematic Layout



9.7.1. DRQ INTERRUPT ROUTER SUMMARY

The internal and external DRQs are set out as per Table 9-17. below. The DRQs are layed out on their priority basis. If there are two or more input lines routed to the same output, only the highest priorty input will be taken into account, the lower inputs will be ignored. The highest priority is placed first and the lowest is last.

Table 9-17. DRQ Routing Control Registers

Access	Default Value after Reset	DRQ	Register Name	Mnemonic
0x428h	00h	DRQ_INT[0]	DRQ Internal Routing Control Register 0	*IDRQ_cont0*
0x429h	00h	DRQ_INT[1]	DRQ Internal Routing Control Register 1	*IDRQ_cont1*
0x42Ah	00h	DRQ_INT[2]	DRQ Internal Routing Control Register 2	*IDRQ_cont2*
0x42Bh	00h	DRQ_INT[3]	DRQ Internal Routing Control Register 3	*IDRQ_cont3*
0x420h	80h	DRQ_EXT[0]	DRQ External Routing Control Register 0	*EDRQ_cont0*
0x421h	81h	DRQ_EXT[1]	DRQ External Routing Control Register 1	*EDRQ_cont1*
0x422h	82h	DRQ_EXT[2]	DRQ External Routing Control Register 2	*EDRQ_cont2*
0x423h	83h	DRQ_EXT[3]	DRQ External Routing Control Register 3	*EDRQ_cont3*
0x424h	84h	DRQ_EXT[4]	DRQ External Routing Control Register 4	*EDRQ_cont4*
0x425h	85h	DRQ_EXT[5]	DRQ External Routing Control Register 5	*EDRQ_cont5*
0x426h	86h	DRQ_EXT[6]	DRQ External Routing Control Register 6	*EDRQ_cont6*
0x427h	87h	DRQ_EXT[7]	DRQ External Routing Control Register 7	*EDRQ_cont7*

The DRQ router is connected between the DRQ-requesting devices and the two 8237 DMA Controller devices. It requires 12 registers, one for each DRQ input (eight external, four internal). These registers are implemented from 420h to 42Ch. Details of all 12 registers are given in Table 9-17.

9.7.2. DRQ ROUTING CONTROL REGISTERS

These four 8-bit registers control the routing of Internal DRQ0 to DRQ3 (DRQ_INT[0] to DRQ_INT[3]) and External Interrupts DRQ0 to DRQ7 (DRQ_EXT[0] to DRQ_EXT[7]) to one of the DRQ inputs of the 8237 as follows:

IDRQ_cont0 Access: see Table 9-17

7	6	5	4	3	2	1	0
RE	Rsv				RC		
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	RE	**Routing Enable.** If set to a ‘1’, this bit enables the routing of the Internal DRQ, otherwise the DRQ is unconnected.
Bits 6-3	Rsv	**Reserved**. Writes have no affect. Reads return undefined value.
Bits 2-0	RC	**Routing Control.** These bits route the IDRQ (see Table 9-18)

Table 9-18. IDRQ Route

Bit 2	Bit 1	Bit 0	Internal DRQ Route
0	0	0	DRQ0
0	0	1	DRQ1
0	1	0	DRQ2
0	1	1	DRQ3
1	0	0	DRQ4
1	0	1	DRQ5
1	1	0	DRQ6
1	1	1	DRQ7

10. IDE CONTROLLER

10.1. INTRODUCTION

The IDE (Integrated Drive Electronics) controller provides two IDE channels, primary and secondary, for interfacing with up to four IDE drives. It supports PIO modes 0 to 4 plus DMA modes 0 to 2. The timings are individually programmable for all four IDE devices. Each channel has a four double-word FIFO for data transfers which allows four levels of write posting or read prefetch. Accesses to the 8-bit non-data IDE registers bypass the FIFOs.

For each of the four drives there are three bits in the configuration registers which can selectively enable write posting, read prefetch and ATAPI read prefetch. If read prefetch is enabled, the IDE controller will prefetch data from the drive after the first read has been made. The prefetching will stop after 256 data reads (512 Bytes), which is the normal sector size. If the current command to the drive is ATAPI packet (A0h), or service (A2h), then the read prefetch will be disabled unless ATAPI read prefetch is set.

The two channels of the IDE controller can be individually programmed to operate in either legacy or native mode. In legacy mode, the IDE interrupts are hardwired to INT 14 & 15. In native mode, they both connect to PCI INTA. If legacy mode is selected, INT 14 & 15 will not be available on the ISA bus even if IDE interrupts are disabled. In legacy mode, the primary and secondary channels are hardwired to IO addresses 1F0h-1F7h / 3F6h and 170h-177h / 376h respectively. In native mode the IO addresses are programmed by configuration registers. For information on PIO mode, please refer to the ATAPI Standard.

The IDE controller provides DMA bus master transfer between IDE devices and system memory, with scatter/gather capability. By performing the IDE data transfer as a bus master, the Bus Master Device off-loads the CPU (no programmed IO for data transfer) and improves system performance in multitasking environments.

Before issuing the DMA command, the system software must first create a Physical Region Descriptor (PRD) table in system memory. This table contains a list of pointers and byte counts for each entry. A register in the IDE controller is set to point to this table. The IDE DMA controller will read from system memory during DMA initialization. Each entry in the PRD table is eight bytes long and will have the format below:

10.2. PRD TABLE ENTRY

PRD1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EOT	Rsv														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOW															Rsv
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bits 31	**OET**	This bit set to '1' if it is the last entry in the table
Bits 30-16	**Rsv**	**Reserved**
Bit 15-1	**NOWS**	**Number of 16-bit data packets**
Bit 0	**Rsv**	**Reserved;** This bit must be set to 0

PRD0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MRPAS															
Default value after reset = undefined															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MRPAS															Rsv
Default value after reset = undefined															

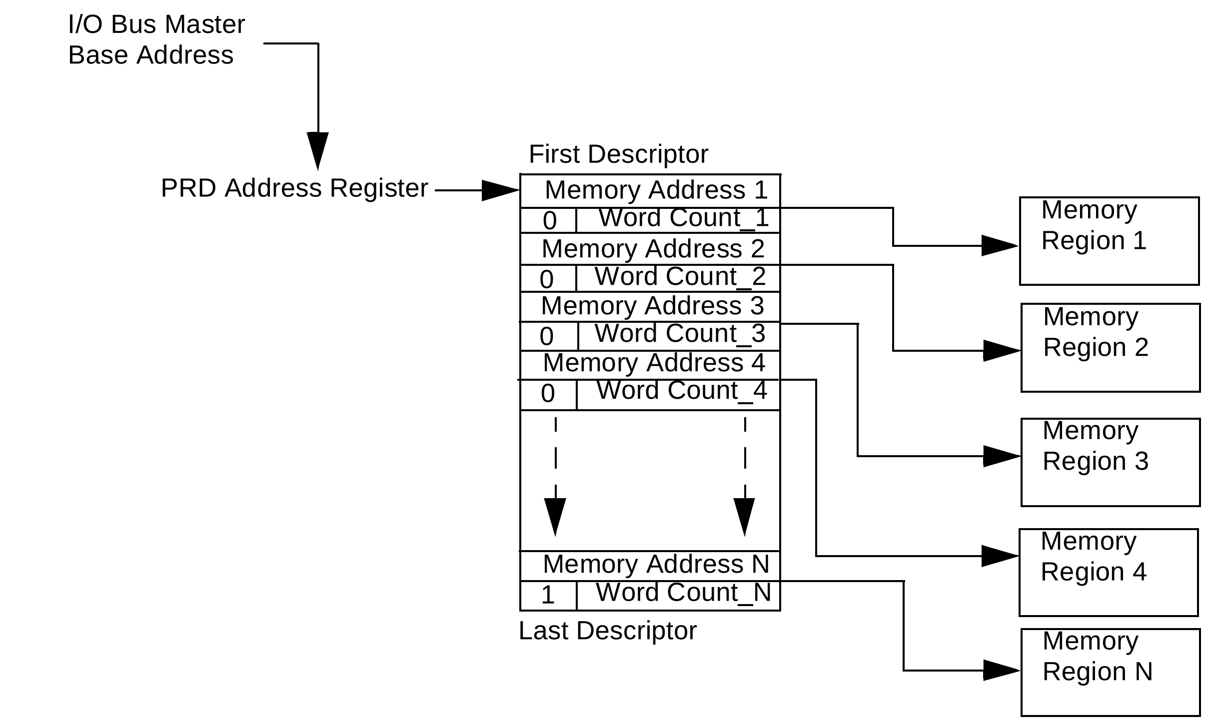
Bit Number	Mnemonic	Description
Bit 31-1	**MRPAS**	**Memory Physical Address of the first descriptor**
Bit0	**Rsv**	**Reserved;** This bit must be set to 0

The table must be aligned on a 4 byte boundary and should not cross a 64k boundary.

A memory region also should not cross a 64k boundary neither. An example of a PRD table is shown in Figure 10-1..

The primary and secondary channels each have a PRD address pointer register.

To save pins, the IDE controller shares pins with the ISA interface. On the IDE data bus, CS1 & CS3 signals are shared with the ISA address bus and keyboard controller/RTC pins. These signals are isolated by external transceiver devices. The ISAOE signal selects whether the pins are in IDE or ISA mode. The



Note: For information on IO Bus Master Base Address, refer to Section 8.8.15. .

Figure 10-1. PRD Table Entry Example

South Bridge arbitrates between the IDE controller and the ISA bus bridge to select which has control of the shared pins.

10.3. IDE BUS MASTER REGISTERS

This document defines a register level programming interface for the internal busmaster ATA-compatible (IDE) disk controller that directly moves data between IDE devices and main memory.

The system using this programming interface will benefit from bundled software shipped with major Operating Systems, limiting the amount of software development required to provide a complete product.

The master mode programming interface is an extension of the standard IDE programming model. This means that devices can always be dealt with using the standard IDE programming model, with the master mode functionality used when the appropriate driver and devices are present. Master operation is designed to work with any IDE device that supports DMA transfers on the IDE bus. Devices that only work in PIO mode can be used through the standard IDE programming model.

The programming interface defines a simple scatter/gather mechanism, allowing large transfer blocks to be scattered to or gathered from memory. This cuts down the number of interrupts to and interactions with the CPU. The interface defined here supports two IDE channels (primary and secondary). Individual controllers that support more than two channels will need to appear to software as multiple controllers if the standard drivers are to be used. Master IDE controllers should default to Mode 0 Multiword DMA timings to ensure operation with DMA capable IDE devices without the need for controller-specific code to initialize controller-specific timing parameters.

10.3.1. PHYSICAL REGION DESCRIPTOR TABLE

Before the controller starts a master transfer, it is given a pointer to a Physical Region Descriptor Table. This table contains some of a number of the Physical Region Descriptors (PRD); these define the memory areas that are involved in the data transfer. The descriptor table must be aligned on a 4 Byte boundary and the table cannot cross a 64 KByte boundary in memory.

10.3.2. PHYSICAL REGION DESCRIPTOR

The physical memory region to be transferred is described by a Physical Region Descriptor (PRD). The data transfer will not proceed until all regions described by the PRDs in the table have been transferred.

Each Physical Region Descriptor entry is eight Bytes long.

- The first four bytes specify the byte address of a physical memory region.
- The next two bytes specify the count of the region in bytes (64K byte limit per region).

A value of zero in these two bytes indicates 64 KByte. Bit 7 of the last byte indicates the end of the table; the Bus Master operation terminates when the last descriptor has been retired.

Note: The memory region specified by the descriptor is further restricted such that the region cannot straddle a 64K boundary. This means that the byte count can be limited to 64K, and the incrementer for the current address register need only extend from bit [1] to bit [15]. Also, the total sum of the descriptor byte counts must be equal to, or greater than, the size of the disk transfer request. If greater than, then the driver must terminate the Bus Master transaction (by resetting bit zero of the command register to zero) when the drive issues an interrupt to signal transfer completion.

10.4. BUS MASTER IDE REGISTER DESCRIPTION

The bus master IDE function uses 16 bytes of IO space. All bus master IDE IO space registers can be accessed as byte, word or Dword quantities. The description of the 16 bytes of IO registers is given in Table 10-5.

Table 10-5. Bus Master IDE Register Description

Offset	Register	R/W Status
00h	Bus Master IDE Command register Primary	R/W
01h	Device Specific	
02h	Bus Master IDE Status register Primary	RWC
03h	Device Specific	
04h-07h	Bus Master IDE PRD Table Address Primary	R/W
08h	Bus Master IDE Command register Secondary	R/W
09h	Device Specific	
0Ah	Bus Master IDE Status register Secondary	RWC
0Bh	Device Specific	
0Ch-0Fh	Bus Master IDE PRD Table Address Secondary	R/W

10.6. BUS MASTER IDE COMMAND REGISTER

10.6.1. IDE COMMAND REGISTER

This 8-bit Register is addressed at offset Base + 00h for the Primary IDE Channel and Base + 08h for the Secondary IDE Channel.

This register enables/disables Bus Master capability for the IDE function and provides direction control for the IDE DMA transfers. This register also provides the bits that software uses to indicate DMA capability of the IDE device.

IDE_COM

7	6	5	4	3	2	1	0
Rsv				RWCOM	Rsv		SSBM
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7-4	**Rsv**	**Reserved**
Bit 3	**RWCOM**	**Read Write Control.** This bit sets the direction of the bus master transfer: when set to zero, PCI bus master reads are performed. When set to one, PCI bus master writes are performed: 0 = PCI bus master read 1 = PCI bus master write While a synchronous DMA transfer is in progress, this bit will be READ ONLY. The bit will return to read/write once the synchronous DMA transfer has been completed or halted. This bit must NOT be changed when the bus master function is active
Bit 2-1	**Rsv**	**Reserved**
Bits 0	**SSBM**	**Stop/Start Bus Master.** Writing a ‘1’ to this bit enables bus master operation of the controller. Bus master operation begins when this bit is detected changing from a zero to a one. The controller will transfer data between the IDE device and memory only when this bit is set. Master operation can be halted by writing a ‘0’ to this bit. All state information is lost when a ‘0’ is written;. Master mode operation cannot be stopped and then resumed. If this bit is reset while bus master operation is still active (i.e., Bit 0= 1 in the Bus Master IDE Status Register for that IDE channel) and the drive has not yet finished its data transfer (bit 2=0 in the channel’s Bus Master IDE Status Register), the Bus Master command is said to be aborted and data transferred from the drive may be discarded before being written to the system memory. This bit is intended to be reset after the data transfer is completed, as indicated by either the Bus Master IDE Active bit or the Interrupt bit of the IDE Status register for that IDE channel being set, or both.

10.6.2. IDE STATUS REGISTER

This 8-bit Register is addressed at offset Base + 02h for the Primary IDE Channel and Base + 0Ah for the Secondary IDE Channel.

This register provides status information about the IDE device and state of the IDE DMA transfer. Table 10-7. describes IDE Interrupt Status and Bus Master IDE Active bit states after a DMA transfer has been started.

The IDE Status Register is illustrated in the following table.

IDE_COM

7	6	5	4	3	2	1	0
SO	D1DMA	D0DMA	Rsv		RWI	RWE	RWMI
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**SO**	**Simplex only.** This is hardwired to'0'.
Bit 6	**D1DMA**	**Drive 1 DMA Capable.** This read/write bit is set by device dependent code (BIOS or device driver) to indicate that drive 1 for this channel is capable of DMA transfers, and that the controller has been initialised for optimum performance.
Bit 5	**D0DMA**	**Drive 0 DMA Capable.** This read/write bit is set by device dependent code (BIOS or device driver) to indicate that drive 0 for this channel is capable of DMA transfers, and that the controller has been initialised for optimum performance.
Bits 4-3	**Rsv**	**Reserved.** These bits return '0' when read.
Bit 2	**RWI**	**Read/Write Interrupt.** This bit is set by the rising edge of the IDE interrupt line. This bit is cleared when a '1' is written to it by software. Software can use this bit to determine if an IDE device has asserted its interrupt line. When this bit is read as a one, all data transferred from the drive is visible in system memory. For further details see Table 10-7.
Bit 1	**RWE**	**Read/Write Error.** This bit is set when the controller encounters an error in transferring data to/from memory. The exact error condition is bus specific and can be determined in a bus specific manner. This bit is cleared when a '1' is written to it by software.
Bit 0	**RWMI**	**Read/Write Bus Master IDE Active.** This bit is set to 1 when bit 0 in the Command register is set to 1. This bit is cleared (set to 0) when the last transfer for a region is performed, where EOT for that region is set in the region descriptor. It is also cleared when the Start bit is cleared in the Command register. When this bit is read as a zero, all data transferred from the drive during the previous bus master command is visible in system memory, unless the bus master command was aborted. For further details seeTable 10-7.

Table 10-7. Interrupt/Activity Status Combinations

Bit 2	Bit 0	Description
0	1	DMA transfer is in progress. No interrupt has been generated by the IDE device.
1	0	The IDE device generated an interrupt and the Physical Region Descriptors exhausted.This is normal completion where the size of the physical memory regions is equal to the IDE device transfer size.
1	1	The IDE device generated an interrupt. The controller has not reached the end of the physical memory regions. This is a valid completion case when the size of the physical memory regions is larger than the IDE device transfer size.
0	0	Error condition. If the IDE DMA Error bit is 1, there is a problem transferring data to/from

10.7.3. DESCRIPTOR TABLE POINTER REGISTER

This 32-bit Register is addressed at offset Base I/O+ 04h for the Primary IDE Channel and Base I/O+ 0Ch for the Secondary IDE Channel.

This register provides the base memory address of the Descriptor Table. The Descriptor Table must be DWord aligned and must not cross a 4-Kbyte boundary in memory.

DT_Point

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BADT															
Default value after reset = 00h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BADT														Rsv	
Default value after reset = 00h															

Bit Number	Mnemonic	Description
Bits 31-2	**BADT**	**Base address of Descriptor table.** This field corresponds to A[31-2].
Bits 1-0	**Rsv**	**Reserved**

The Descriptor Table must be Dword aligned. The Descriptor Table must not cross a 64K boundary in memory.

10.8. OPERATION

10.8.1. STANDARD PROGRAMMING SEQUENCE

To initiate a bus master transfer between memory and an Hard Disk device, the following steps are required:

1) Software prepares a PRD Table in system memory. Each PRD is 8 bytes long and consists of an address pointer to the starting address and the transfer count of the memory buffer to be transferred. In any given PRD Table, two consecutive PRDs are offset by 8-bytes and are aligned on a 4-byte boundary.

2) Software provides the starting address of the PRD Table by loading the PRD Table Pointer Register. The direction of the data transfer is specified by setting the Read/Write Control bit. Clear the Interrupt bit and Error bit in the Status register.

3) Software issues the appropriate DMA transfer command to the disk device.

4) Engage the bus master function by writing a ‘1’ to the Start bit in the Bus Master IDE Command Register for the appropriate channel.

5) The controller transfers data to/from memory responding to DMA requests from the IDE device.

6) At the end of the transfer the IDE device signals an interrupt.

7) In response to the interrupt, software resets the Start/Stop bit in the command register. It then reads the controller status and then the drive status to determine if the transfer completed successfully.

10.9. DATA SYNCHRONIZATION

When reading data from an IDE device, that data may be buffered by the IDE controller before using a master operation to move the data to memory. The IDE device driver in conjunction with the IDE controller is responsible for guaranteeing that any buffered data is moved into memory before the data is used.

The IDE device driver is required to do a read of the controller Status register after receiving the IDE interrupt. If the Status register returns with the Interrupt bit set, then the driver knows that the IDE device generated the interrupt (important for shared interrupts) and that any buffered data has been flushed to memory. If the Interrupt bit is not set, then the IDE device did not generate the interrupt and the state of the data buffers is unknown.

When the IDE controller detects a rising edge on the IDE device interrupt line (INTRQ), it is required to:

- Flush all buffered data
- Set the Interrupt bit in the controller Status register
- Guarantee that a read to the controller Status register does not complete until all buffered data has been written to memory.

Another way to view this requirement is that the first read to the controller Status register in response to the IDE device interrupt must return with the Interrupt bit set and with the guarantee that all buffered data has been written to memory.

10.9.1. STATUS BIT INTERPRETATION

The table below gives a description of how to interpret the Interrupt and Active bits in the Controller status register after a DMA transfer has been started.

Interrupt bit	Active bit	Description:
0	1	DMA transfer is in progress. No interrupt has been generated by the IDE device.
1	0	The IDE device generated an interrupt. The controller exhausted the Physical Region Descriptors. This is the normal completion case where the size of the physical memory regions was equal to the IDE device transfer size.
1	1	The IDE device generated an interrupt. The controller has not reached the end of the physical memory regions. This is a valid completion case where the size of the physical memory regions was larger than the IDE device transfer size.
0	0	This bit combination signals an error condition. If the Error bit in the status register is set, then the controller has some problem transferring data to/from memory. Specifics of the error have to be determined using bus-specific information. If the Error bit is not set, then the PRD's specified a smaller size than the IDE transfer size.

10.10. ERROR CONDITIONS

IDE devices are sector based mass storage devices. The drivers handle errors on a sector by sector basis; either a sector is transferred successfully or it is not. If the IDE DMA slave device never completes the transfer due to a hardware or software error, the Bus Master IDE command will eventually be stopped (by setting Command Start bit to zero) when the driver times out the disk transaction. Information in the IDE device registers will help isolate the cause of the problem.

If the controller encounters an error while doing the bus master transfers, it will stop the transfer (i.e. reset the Active bit in the Command register) and set the ERROR bit in the Status register. The controller does not generate an interrupt when this happens. The device driver can use device specific information (e.g. PCI Configuration Space Status register) to determine what caused the error.

Whenever a requested transfer does not complete properly, information in the IDE device registers (Sector Count) can be used to determine how much of the transfer was completed and to construct a new PRD table to complete the requested operation. In most cases the existing PRD table can be used to complete the operation.

10.11. PCI SPECIFICS

Bus master IDE controllers built to attach to a PCI bus must have the following characteristics:

1) The Class Code in PCI configuration space indicates IDE device and bit 7 of the Programming Interface register (offset 0x09) in PCI configuration space must be set to 1 to indicate that the device supports the Master IDE capability.

2) The control registers for the controller are allocated via the devices Base Address register at offset 0x20 in PCI configuration space.

3) In the controller Status register the Error bit will be set and the Active bit reset if any of the following conditions occur on the PCI bus while the controller is doing a master operation on the bus. The exact cause can be determined by examining the Configuration Space Status register.

Error Condition	Configuration Space Status bits
Target Abort	Any time bit 12 of the Config Space Status register is set.
Master Abort	Any time bit 13 of the Config Space Status register is set.
Data Parity	Any time bit 8 of the Config Space Status register is set.
Error Detected	

11. VGA CONTROLLER

11.1. INTRODUCTION

The STPC integrates a full VGA Controller with Extended Functions together with a Colour Digital to Analog output (RAMDAC) and a Graphics Engine. The VGA Controller provides the basic video display function. It generates the timing and logic required to create an output data stream from the video buffer and the appropriate horizontal and vertical synchronisation pulses.

The on-chip triple RAMDAC runs at up to 135 MHz, using an external frequency synthesiser, allowing a display up to 1280 x 1024 at 75 Hz. Colour is handled using 8-bits, 16-bits, 24-bits or 32-bits per pixel. VDU Graphics standards can be read through the on-chip Display Data Channel (DDC) link.

11.2. VGA CONTROLLER

The VGA controller of the STPC is 100% backward compatible with the VGA standard specification. In addition, enhancements made to the VGA standard are detailed in the following sub-sections.

Resolutions of up to 1024 x 768 and colour depths of 8, 16, 24 and 32 bits per pixel are supported. The integrated RAMDAC supports digital to analog conversion rates up to 135 MHz. This along with peak video bandwidth of 320 MBytes/sec enables the VGA controller to support 1024 x 768 x24 and 800 x 600 x 32 resolutions at 75 Hz refresh rate.

To support vertical resolutions up to 1024 pixels, vertical timing parameters have been extended from 10 bits to 11 bits. The VGA defined horizontal timing parameters are compatible with the above resolutions. The horizontal and vertical timing counters and the sync and blank generation logic operate synchronously to DCLK which can be up to 135 MHz in frequency.

Pixel colour depths are specified by programming the Palette Control register (CR28) appropriately. Eight-bit colour modes use the RAMDAC look-up table to form 18-bit or 24-bit colours. All other modes bypass the look-up table and drive the DACs directly.

The Graphics Core is capable of using up to 4 Mb of available memory as its frame buffer. The Cathode Ray Tube Controller (CRTC) Start Address uses 20-bits to allow for locating the frame buffer at any double word boundary within this 4 Mb of memory. This frame buffer sits within the 16 MBytes Graphics buffer area. Refer to the Graphics Engine Section for further details on the Graphics Memory Architecture.

Video data is automatically extracted from the frame buffer by the CRTC. A FIFO structure ensures that the video display is continually refreshed without loss of data and visual artifacts. Independent high and low level watermarks can be programmed to accelerate or decelerate the demands on the memory arbitration logic.

The CRTC can be programmed to support interlaced monitors and timings. It also supports hardware generated cursor in text mode and a 64 x 64 bit cursor in Graphics modes. This graphics mode cursor is software programmable with separate programmable XOR and AND masks in memory.

If an external add-in VGA card is placed in the system, the on-chip VGA controller can be disabled in order to work with this external card. It is possible to enable / disable the system back to dual use VGA controller if necessary.

11.3. VGA REGISTERS

The following sections describe both the standard VGA compatible register definitions and the definitions of register extensions specific to the STPC VGA controller.

The ‘X’ within some IO addresses represents a ‘B’ if monochrome operation is enabled and a ‘D’ if colour operation is in effect.

11.4. GENERAL VGA REGISTERS

11.4.1. MOTHERBOARD ENABLE REGISTER (RW)

MBEN Access = 0x094h Regoffset =

7	6	5	4	3	2	1	0
Rsv		ME	Rsv	MBEN	Rsv		
Default value after reset = 28h							

Bit Number	Mnemonic	Description
Bits 7-6	**Rsv**	**Reserved**, read as ‘0’s.
Bits 5	**ME**	**Motherboard Enable.** If the VGA is configured to operate on the motherboard, then when this bit is set to ‘0’, it allows read and write access to port 102h. All other IO and memory accesses are ignored. When set to a ‘1’, this bit allows access to all IO and memory, but access to port 102h is ignored.
Bits 4	**Rsv**	**Reserved**, reads as ‘0’.
Bits 3	**MBEN**	**MBEN Video System Enable**. When ‘0’ this bit disables all IO and memory accesses to the VGA as well as the DAC registers. Accesses to 94h remain enabled. When ‘1’, Video system enable bits of port 0102h and 03C3h determine the accessibility of the VGA. The VGA continues to display video data while disabled.
Bits 2-0	**Rsv**	**Reserved**, read as ‘0’s.

Programming notes:

The contents of this register are not altered by drawing operations.

11.4.2. ADD-IN VGA ENABLE REGISTER (RW)

ADDEN Access = 0x46E8h Regoffset =

7	6	5	4	3	2	1	0
Rsv			AE	ADDEN VSE	Rsv		
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-5	**Rsv**	**Reserved**, read as 0's.
Bits 4	**AE**	**Addin Enable.** If the VGA is configured to operate on an add-in card, then when this bit is set to '1', it allows read and write access to port 102h. All other IO and memory accesses are ignored. When set to a '0', this bit allows access to IO and memory, but access to port 102h is ignored.
Bits 3	**ADDEN VSE**	**ADDEN Video System Enable.** When '0' this bit disables all IO and memory accesses to the VGA as well as the DAC registers. Accesses to 46E8h remain enabled. When '1', Video system enable bits of port 0102h determine the accessibility of the VGA. The VGA continues to display video data while disabled.
2-0	**Rsv**	**Reserved,** read as '0's.

Programming notes:

The contents of this register are not altered by drawing operations.

11.4.3. VIDEO SUBSYSTEM ENABLE 1 REGISTER (RW)

VSE1 Access = 0x102h Regoffset =

7	6	5	4	3	2	1	0
Rsv							VSE 1
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-1	**Rsv**	**Reserved**, read as '0's.
Bit 0	**VSE 1**	**Video System Enable.** When '0', this bit disables all IO and memory accesses to the VGA as well as DAC registers except port 102h. Port 102h remains accessible to allow enabling of the VGA. Ports 46E8h and 94h are also not affected by this bit. The VGA continues to display video data while disabled.

Programming notes:

The contents of this register are not altered by drawing operations.

11.4.4. VIDEO SUBSYSTEM ENABLE 2 REGISTER (RW)

VSE2 Access = 0x3C3h Regoffset =

7	6	5	4	3	2	1	0
Rsv							VSE 2
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-1	**Rsv**	**Reserved**, read as ’0’s.
Bit 0	**VSE 2**	**Video System Enable.** When ‘0’, this bit disables all IO and memory accesses to the VGA as well as DAC registers except ports 102h and 3C3h. Ports 102h and 03C3h remain accessible to allow enabling of the VGA. Port 94h is also not affected by this bit. The VGA continues to display video data while disabled.

Programming notes:

The contents of this register are not altered by drawing operations.

11.4.5. MISCELLANEOUS OUTPUT REGISTER (RW)

MISC			Access = 0x3CCh/0x3C2h				Regoffset =
7	6	5	4	3	2	1	0
VRP	HRP	OEPS	Rsv	CS		E RAM	IO A
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**VRP**	**Vertical retrace polarity.** 0: Active high 1: Active low
Bit 6	**HRP**	**Horizontal retrace polarity**. 0: Active high 1: Active low For older IBM compatible colour monitors, the polarity of the vertical and horizontal retrace pulses was used to define the vertical scan rate, as given inTable 11-1.
Bit 5	**OEPS**	**Odd/Even Page Select.** This bit selects between two 64 K pages of memory (of a 128 K plane) when the VGA is in odd/even mode (replaces the least significant bit of the memory address). ‘0’ = low 64 K page. ‘1’ = high 64 K page. This bit is only effective in Mode 0, 1, 2, 3, or 7.
Bit 4	**Rsv**	**Reserved**, reads as ‘0’.
Bits 3-2	**CS**	**Clock Selects.** Selects one of the four synthesiser pairs when DCLK source is onchip PLL’s.
Bit 1	**E RAM**	**Enable RAM.** When ‘0’, this bit disables host accesses to the display RAM. The access to the ROM, however, remains enabled. Setting this bit to ‘1’ enables accesses to the display buffer.
Bit 0	**IO A**	**IO Address.** This bit defines the address map of the following registers (see Table 11-2).

Table 11-1. Vertical and Horizontal Polarities

Bit7	Bit6	Active Lines	Vertical Total
0	0	Reserved	Reserved
0	1	400 lines	414 lines
1	0	350 lines	362 lines
1	1	480 lines	496 lines

Table 11-2. IO Address

Register	Bit 0 = ’0’	Bit 0 = ’1’
CRTC Registers	03BXh	03DXh
Input #1 Register	03BAh	03DAh

Programming notes:

The contents of this register are not altered by drawing operations.

11.4.6. INPUT STATUS REGISTER #0 (R)

INP0 Access = 0x3C2h Regoffset =

7	6	5	4	3	2	1	0
VRF	Rsv		R S	Rsv			
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**VRF**	**Vertical Retrace Flag.** This bit is set at the beginning of the vertical retrace period if bit 4 of CR11 (Vertical Retrace End register) is set to one. Once set, this bit is cleared when bit 4 of CR11 is reset to 0. This recording of the vertical retrace interrupt is independent of bit 5 (disable vertical interrupt) of CR11. See the description of CR11 for more details.
Bits 6-5	**Rsv**	**Reserved.** These bits read as ones.
Bit 4	**RS**	**RAMDAC Sense.** This bit is connected to the SENSE signal of the RAMDAC. It is used by the BIOS to auto-detect the monitor type.
Bits 3-0	**Rsv**	**Reserved.** These bits read as zero.

11.4.7. INPUT STATUS REGISTER *#1 (R)*

INP1				Access = 0x3XAh			Regoffset =
7	6	5	4	3	2	1	0
Rsv		DU		VR	Rsv		R
Default value after reset =							

Bit Number	Mnemonic	Description
Bit 7-6	**Rsv**	**Reserved.** These bits read as zero.
Bits 5-4	**DU**	**Diagnostic Use.** These bits reflect 2 of the 8 bit video output data during display periods and overscan colour data during non-display periods. Selection of one of four pairs of bits is controlled by bits 5-4 of the AR12 as in Table 11-3.
Bit 3	**VR**	**Vertical Retrace.** A one in this position indicates that a vertical retrace is in progress.
Bits 2-1	**Rsv**	**Reserved.** Bit 2 reads as one; bit 1 reads as zero.
Bit 0	**R**	**Retrace.** A one in this position indicates that a horizontal OR vertical retrace is in progress.

Table 11-3. Video Output Data Selection

AR12		**Diagnostic Bits**	
Bit 5	**Bit 4**	**Bit 5**	**Bit 4**
0	0	Video2	Video0
0	1	Video5	Video4
1	0	Video3	Video1
1	1	Video7	Video6

11.5. VGA SEQUENCER REGISTERS

11.5.1. SEQUENCER INDEX REGISTER (RW)

SRX Access = 0x03C4h/0x03C5h Regoffset =

7	6	5	4	3	2	1	0
Rsv					SI		
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-4	**Rsv**	**Reserved**, reads as 0’s.
Bit 3		
Bits 2-0	**SI**	**Sequencer Index.** These bits point to the register that is accessed by the next read or write to port 03C5h.

Programming notes:

The contents of this register are not altered by drawing operations.

11.5.2. SEQUENCER RESET REGISTER (RW)

SR0			Access = 0x03C4h/0x03C5h				Regoffset = 000h
7	6	5	4	3	2	1	0
Rsv						SR	AR
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-2	**Rsv**	**Reserved**, reads as ’0’s.
Bit 1	**SR**	**Synchronous Reset.** When set to ‘0’ terminates display memory accesses. This bit, as well as bit 0 of this register, must be set to ‘1’ to enable sequencer operations. The Clocking Mode register (SR1) bits 0 and 3, and Miscellaneous Output register bits 2-3 must not be changed unless this bit is set to ‘0’ to avoid loss of memory contents.
Bit 0	**AR**	**Asynchronous Reset.** This bit performs the same function as bit 1 except when set from ‘1’ to ‘0’, it also clears the Character Map select register (SR3) to ‘0’.

Programming notes:

The contents of this register are not altered by drawing operations.

11.5.3. SEQUENCER CLOCKING MODE REGISTER (RW)

SR1 Access = 0x03C4h/0x03C5h Regoffset = 001h

7	6	5	4	3	2	1	0
Rsv		SO	S4	DC	SL	Rsv	8/9 DC
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-6	**Rsv**	**Reserved**, reads as '0's.
Bit 5	**SO**	**Screen Off.** Setting this bit to '1' blanks the screen by driving black colour (not overscan) on the screen. This facilitates the CPU to access video memory at maximum possible bandwidth.
Bit 4	**S4**	**Shift4.** Together with Shift Load (bit 2), this bit controls the loading of the video serialisers, as in Table 11-4.
Bit 3	**DC**	**Dot Clk.** When '0' sets the dot clock to be the same as the input dot clock. When '1', divides the input dot clock by 2 to derive the dot clock. The input dot clock is divided by 2 for 320 and 360 horizontal pixel modes 0, 1, 4, 5, D and 13. This is can not be used when using an external DCLK
Bit 2	**SL**	**Shift Load**, see Bit 4 - Shift4.
Bit 1	**Rsv**	**Reserved,** reads as '0'.
Bit 0	**8/9 DC**	**8/9 Dot Clock.** When '0', this bit causes the character clock to be 9 dots wide. When '1', an 8-dot wide character clock is selected.

Table 11-4. Video Serialiser Load Clock

Bit4	Bit2	Video Serialiser Load clock	Resolution
0	0	Every character	720 dots/line
0	1	Every second character	360 dots/line
1	X	Every fourth character	180 dots/line

Programming notes:

The contents of this register are not altered by drawing operations.

11.5.4. SEQUENCER PLANE MASK REGISTER (RW)

SR2 Access = 0x03C4h/0x03C5h Regoffset = 002h

7	6	5	4	3	2	1	0
Rsv				EP3	EP2	EP1	EP0
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-4	**Rsv**	**Reserved**, reads as ’0’.
Bit 3	**EP3**	**Enable Plane 3**, Write enable for plane 3. A ‘0’ in this bit disables writes to plane 3.
Bit 2	**EP2**	**Enable Plane 2**
Bit 1	**EP1**	**Enable Plane 1**
Bit 0	**EP0**	**Enable Plane 0**

The planes are used in different manners by the various modes. These are shown in Table 11-5.

Table 11-5. Various Modes

Mode	Plane 0	Plane 1	Plane 2	Plane 3
Text Modes 0,1, 2, 3, 7	Character Data	Attribute Data	Font Data	Unused
16-bit Colour Graphics Modes D, E, 10, 12	Pixel Bit 0	Pixel Bit 1	Pixel Bit 2	Pixel Bit 3
4-Colour Mono Graphics Mode F	Video	Ignored	Intensity	Ignored
4-Colour Modes 4, 5	Even Byte	Odd Byte	Unused	Unused
2-Colour Mono Graphics Mode 6	Even Byte	Odd Byte	Unused	Unused
2-Colour Mono Graphics Mode 11	All Bytes	Unused	Unused	Unused
256-Colour Graphics Mode 13	Byte 0	Byte 1	Byte 2	Byte 3

Programming notes:

The contents of this register are not altered by drawing operations.

11.5.5. SEQUENCER CHARACTER MAP REGISTER (RW)

SR3 Access = 0x03C4h/0x03C5h Regoffset = 003h

5	4	3	2	1	0
SFB	PFB	SFB	SFB	PFB	PFB
Default value after reset = 00h					

Bit Number	Mnemonic	Description
Bit 5	**SFB**	**Secondary Font Block Select bit 0.**
Bit 4	**PFB**	**Primary Font Block Select bit 0.**
Bit 3	**SFB**	**Secondary Font Block Select bit 2.**
Bit 2	**SFB**	**Secondary Font Block Select bit 1.**
Bit 1	**PFB**	**Primary Font Block Select bit 2.**
Bit 0	**PFB**	**Primary Font Block Select bit 1.**

Programming notes:

Used in text mode to select the primary and secondary font tables when the attribute bit 3 is ‘0’ (for primary) or ‘1’ (for secondary) as given inTable 11-6 and Table 11-7.

Table 11-6. Primary Font

Bit 1	Bit 0	Bit 4	Font block #	Table Location
0	0	0	0	0K
0	0	1	4	8 K
0	1	0	1	16 K
0	1	1	5	24 K
1	0	0	2	32 K
1	0	1	6	40 K
1	1	0	3	48 K
1	1	1	7	56 K

Table 11-7. Secondary Font

Bit 3	Bit 2	Bit 5	Font block #	Table Location
0	0	0	0	0 K
0	0	1	4	8 K
0	1	0	1	16 K
0	1	1	5	24 K
1	0	0	2	32 K
1	0	1	6	40 K
1	1	0	3	48 K
1	1	1	7	56 K

This register is reset to ‘0’ by the asynchronous reset via SR0 register.

The contents of this register are not altered by drawing operations.

11.5.6. SEQUENCER MEMORY MODE REGISTER (RW)

SR4				Access = 0x03C4h/0x03C5h			Regoffset = 004h
7	6	5	4	3	2	1	0
Rsv				C4 A	OE	EM	Rsv
Default value after reset = 04h							

Bit Number	Mnemonic	Description
Bits 7-4	**Rsv**	**Reserved**, reads as ’0’s.
Bit 3	**C4 A**	**Chain-4 Addressing.** When set to ‘1’, this bit forces the two least significant host address bits to select the display buffer plane to be accessed by a host read or write. HA1-0 = ‘00’ selects plane 0, HA1-0 = ‘01’ selects plane 1, etc. For writes, the plane selected by the two address bits still must be enabled via the Plane Mask Register (SR2) in order for the writes to take place. During read transfers, when this bit is set to ‘1’, the Graphics Control Read Map register (GR4) is ignored and the Byte from the plane selected by the two least significant host address bits is returned.
Bit 2	**OE**	**Odd/Even# Addressing.** Similar to the Chain-4 bit in that when set to ‘0’ forces the least significant host address bit to select two of the four display planes for host transfers. HA0 = ‘0’ selects planes 0 and 2, and HA1 = ‘1’ selects planes 1 and 3. Selected planes are ANDed with the Plane Mask register (SR2) to generate the plane write enables during write transfers. Read transfers use Map Select bit 1 from GR4 along with HA0 to select one of the 4 Bytes to be returned to the host. Read Map select bit 0 is not used when odd/even addressing mode is enabled.
Bit 1	**EM**	**Extended Memory.** When this bit is ‘0’ it indicates 64K of display memory is present. When ‘1’, indicates that 256K of display memory is present.
Bit 0	**Rsv**	**Reserved,** reads as ‘0’.

Programming notes:

The contents of this register are not altered by drawing operations.

11.5.7. EXTENDED REGISTER LOCK/UNLOCK REGISTER (RW)

SR6 Access = 0x03C4h/0x03C5h Regoffset = 006h

7	6	5	4	3	2	1	0
ER LU							
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-0	**ER LU**	**Extended Registers Lock/Unlock.** When written to with 57h, all extended registers are unlocked. When written to with any value other than 57h, all extended registers are locked. When the extended registers are in the locked state, reads to this register return a zero. When the extended registers are in the unlocked state, reads to this register return a ‘1’.

Programming notes:

The contents of this register are not altered by drawing operations.

11.6. GRAPHICS CONTROLLER REGISTERS

11.6.1. GRAPHICS CONTROLLER INDEX REGISTER (RW)

GRX			Access = 0x03CEh/0x03CFh				Regoffset =
7	6	5	4	3	2	1	0
Rsv				GCI			
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-4	**Rsv**	**Reserved**, reads as ’0’s.
Bits 3-0	**GCI**	**Graphics Controller Index.** These bits point to the register that is accessed by the next read or write to port 03CFh.

Programming notes:

The contents of this register are not altered by drawing operations.

11.6.2. GRAPHICS SET/RESET REGISTER (RW)

GR0 Access = 0x03CEh/0x03CFh Regoffset = 000h

7	6	5	4	3	2	1	0
Rsv				GCSR			
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-4	**Rsv**	**Reserved**, reads as ’0’s.
Bits 3-0	**GCSR**	**Graphics Controller Set/Reset.** These bits define the value written to the four memory planes. In Write Mode 0, only the planes enabled by the Enable Set/Reset Register (GR1) are written to. In Write Mode 3, the contents of the Set/Reset register are always enabled. Bit 0 corresponds to memory plane 0, bit 1 to memory plane 1, etc.

Programming notes:

The contents of this register are not altered by drawing operations.

11.6.3. GRAPHICS ENABLE SET/RESET REGISTER (RW)

GR1 Access = 0x03CEh/0x03CFh Regoffset = 001h

7	6	5	4	3	2	1	0
Rsv				GCSR			
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-4	**Rsv**	**Reserved**, reads as ’0’s.
Bits 3-0	**GCSR**	**Graphics Controller Enable Set/Reset.** These bits define which memory planes are to be written to with the value of the corresponding Set/Reset Register (GR0) in Write Mode 0. In Write Mode 3, the Enable Set/Reset register has no affect. Bit 0 corresponds to memory plane 0, bit 1 to memory plane 1, etc.

Programming notes:

The contents of this register are not altered by drawing operations.

11.6.4. GRAPHICS COLOUR COMPARE REGISTER (RW)

GR2 Access = 0x03CEh/0x03CFh Regoffset = 002h

7	6	5	4	3	2	1	0
Rsv				GCCCR			
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-4	**Rsv**	**Reserved**, reads as '0'.
Bits 3-0	**GCCCR**	**Graphics Controller Colour Compare Register.** These bits are compared with the 4-bit colour of up to 8 pixels in Read Mode 1. The 8-bit (1-bit per pixel) result of the comparison is returned to the host. (A bit of '1' is returned for a match, and '0' for a non-match.) Only those bits enabled by the Colour Don't Care Register (GR7) are matched.

Programming notes:

The contents of this register are not altered by drawing operations.

11.6.5. RASTER OP/ROTATE COUNT REGISTER (RW)

GR3 Access = 0x03CEh/0x03CFh Regoffset = 003h

4	3	2	1	0
GCRO		GCRC		
Default value after reset = undefined				

Bit Number	Mnemonic	Description
Bits 4-3	**GCRO**	**Graphics Controller Raster Op.** These bits define the logical operation to apply to the Host data with the data in the Graphics Controller data latch. The possible values of this field are shown inTable 11-8.
Bits 3-0	**GCRC**	**Graphics Controller Rotate Count.** These bits specify the number of bits that the Host data is rotated before the Raster Op is applied. A count of 0 passes the data through unmodified, a count of 1 rotates the Host data 1 bit to the right.

Table 11-8. Raster Operation

Bit 4	Bit 3	Raster Operation
0	0	NOP - Host data passes through unmodified
0	1	Logical AND of Host and latched data
1	0	Logical OR of Host and latched data
1	1	Logical XOR of Host and latched data

Programming notes:

The contents of this register are not altered by drawing operations.

11.6.6. GRAPHICS READ MAP SELECT REGISTER (RW)

GR4 Access = 0x03CEh/0x03CFh Regoffset = 004h

7	6	5	4	3	2	1	0
Rsv						GCRMS	
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-2	**Rsv**	**Reserved**, reads as ’0’.
Bits 1-0	**GCRMS**	**Graphics Controller Read Map Select.** These bits define the memory plane from which the CPU reads data in Read Mode 0. A value of ‘00’ selects plane 0, ‘01’ selects plane 1, etc. This field also selects one of the 4 Bytes of the Graphics Control Read Data latches.

Programming notes:

The contents of this register are not altered by drawing operations.

11.6.7. GRAPHICS MODE REGISTER (RW)

GR5 Access = 0x03CEh/0x03CFh Regoffset = 005h

7	6	5	4	3	2	1	0
Rsv	SM		OE	RM	Rsv	WM	
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved**, reads as ’0’.
Bits 6-5	**SM**	**Shift mode** These values are given inTable 11-9.
Bit 4	**OE**	**OddEven.** This bit performs no function. It is, however, readable and writable.
Bit 3	**RM**	**Read Mode.** If this bit is set to ‘0’, a host read transfer returns the data Byte corresponding to the plane selected by the Read Map Select Register (GR4). This is also called Read Mode 0. When this bit is set to ‘1’, a host read transfer returns the result of the logical comparison between the data in the four planes selected by the Colour Don’t Care Register (GR7) and the contents of the Colour Compare Register (GR2). This is also called Read Mode 1.
Bit 2	**Rsv**	**Reserved**, reads as ‘0’.
Bits 1-0	**WM**	**Write Mode.** These bits select the write mode as follows inTable 11-10.

Table 11-9. Shift Register Behaviour

Bit 6	Bit 5	Shift Register Behaviour
1	X	The shift registers are loaded in the manner to support 256 colours. This bit should be set to “1” for mode 13 operation.
0	1	2-bit packed pixel mode (modes 4 and 5) support. The data in the four serial shift registers are formatted as ATR0-3.
0	0	Normal shift mode. M0d7-0, M1d7-0, M2d7-0 and M3d7-0 are shifted out with address to the Attributed Controller.

2-bit packed pixel modes:

ATR0: M1d0 M1d2 M1d4 M1d6 M0d0 M0d2 M0d4 M0d6

ATR1: M1d1 M1d3 M1d5 M1d7 M0d1 M0d3 M0d5 M0d7

ATR2: M3d0 M3d2 M3d4 M3d6 M2d0 M2d2 M2d4 M2d6

ATR3: M3d1 M3d3 M3d5 M3d7 M2d1 M2d3 M2d5 M2d7

Table 11-10. Write Behaviour

Bit 1	Bit 0	Write Behaviour
0	0	Write Mode 0
0	1	Write Mode 1
1	0	Write Mode 2
1	1	Write Mode 3

Where:

Write Mode 0: each of the four display memory planes are written with the host data rotated by the rotate count value specified in GR3.
If the Enable Set/Reset register (GR1) enables any of the four planes, the corresponding plane is written with the data stored in the Set/Reset register (GR0). The raster operation specified in GR3 and the bit mask register (GR8) contents alter data being written.

Write Mode 1: each of the four display memory planes are written with the data from the Graphics Controller read data latches. These latches should be loaded by the host via a previous read. The Raster Operation, Rotate Count, Set/Reset Data, Enable Set/Reset and Bit Mask registers have no effect.

Write Mode 2: memory planes 3-0 are filled with the value of the host data bits 3-0, respectively. Data on the host bus is treated as the colour value. The Bit Mask register (GR8) is effected in this mode. A “1” in a bit position in the Bit Mask register sets the corresponding pixel in the addressed Byte to the colour specified by the host data bus. A “0” set the corresponding pixel in the addressed Byte to the corresponding pixel in the Graphics Controller read latches. The Set/Reset, Enable Set/Reset and Rotate Count register have no effect.

Write Mode 3: each of the four video memory planes is written with 8-bits of the colour value contained in the Set/Reset register for that plane. The Enable Set/Reset register as no effect, all bits are enabled. The host data is rotated and ANDed with the Bit Mask register to form an 8-bit value that performs the same function as the Bit Mask register in Write Modes 0 and 2. This write mode can be used to fill an area with a single colour or pattern.

Programming notes:

The contents of this register are not altered by drawing operations.

11.6.8. GRAPHICS MISCELLANEOUS REGISTER (RW)

GR6				Access = 0x03CEh/0x03CFh			Regoffset = 006h
7	6	5	4	3	2	1	0
Rsv				MM		C	GM
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-4	**Rsv**	**Reserved,** read as '0'.
Bits 3-2	**MM**	**Memory Map**. These bits specify the map of the display memory buffers in the CPU address space. They are defined as follows in Table 11-11.
Bit 1	**C**	**Chain2.** This bit performs no function. It is, however, readable and writable.
Bit 0	**GM**	**Graphics Mode.** When this bit is set to '1' graphics mode is selected; otherwise when set to '0' alphanumeric mode is selected. This bit is duplicated in AR10[0].

Table 11-11. Address Map

Bit 3	Bit 2	Address Map
0	0	A0000h to BFFFFh (128K)
0	1	A0000h to BFFFFh (128K)
1	0	B0000h to B7FFFh (32K)
1	1	B8000h to BFFFFh (32K)

Programming notes:

The contents of this register are not altered by drawing operations.

11.6.9. GRAPHICS COLOUR DON'T CARE REGISTER (RW)

GR7 Access = 0x03CEh/0x03CFh Regoffset = 007h

7	6	5	4	3	2	1	0
Rsv				DCPS			
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-4	**Rsv**	**Reserved,** read as '0'.
Bits 3-0	**DCPS**	**Dont_care Colour Plane Selects.** One bit per plane determine whether the corresponding colour plane becomes a don't care when a CPU read from the video memory is done in Read Mode 1. A '1' makes the corresponding plane a don't care plane.

Programming notes:

The contents of this register are not altered by drawing operations.

11.6.10. GRAPHICS BIT MASK REGISTER (RW)

GR8			Access = 0x03CEh/0x03CFh				Regoffset = 008h
7	6	5	4	3	2	1	0
BM							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	**BM**	**Bit Mask.** Any bit programmed to a '0' in this register will cause the corresponding bit in each of the four memory planes to be left unchanged by all operations. The data written into memory in this case will be the data which was read in the previous read operation and stored in the Graphics Controller's read latch. The bit mask is applicable to any data written by the host. The bit mask applies to all four planes simultaneously.

Programming notes:

The contents of this register are not altered by drawing operations.

11.7. ATTRIBUTE CONTROLLER REGISTERS

11.7.1. ATTRIBUTE CONTROLLER INDEX (RW)

The Attribute Controller Index register is used to index into the Attribute Data register array.

Port 3C0h is used for write access to both this index register and, in a subsequent write to this address, to the data register pointed to by the index. There is a flip-flop which changes state after each write to this port. The state of the flip-flop determines whether the next IO write to 3C0h will be to the index register or to a data register. The flip-flop may be initialised - to point to the index register - by performing a read from Input Status Register #1 (IO address 3XAh).

ARX Access = 0x3C0h Regoffset =

7	6	5	4	3	2	1	0
Rsv		PAS	ACI				
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-6	**Rsv**	**Reserved.** Must be written as zero.
Bit 5	**PAS**	**Palette Address Source.** When set to zero, allows host write access to the Attribute Palette Registers. The CRT display is turned off while this bit stays zero and overscan colour is displayed. Setting this bit to one allows normal video pixel display and disables host write access to the Palette registers.

Bits 4-0	**ACI**	**Attribute Controller Index.** Points to the data register which will be accessed by the next write to port 3C0h or the next read from port 3C1h.
		A sample program could be as follows:
		mov DX, 3DAh
		in AL, DX
		mov AL, Index
		mov DX, 3C0h
		out DX, AL
		mov AL, Data
		out DX, AL

11.7.2. ATTRIBUTE PALETTE REGISTERS (RW)

These sixteen registers provide one level of indirection between the colour data stored in the display frame buffer and the displayed colour on the CRT screen. In all modes except 256 colour mode, the (maximum) 4-bit raw colour values select one of these sixteen Palette registers. The 6-bit output of the Palette registers is combined with bits 3-2 of AR14 to form the 8-bit output of the VGA controller. In addition, bits 5-4 of the VGA output may come from either Palette register bits 5-4 or from AR14 bits 1-0 depending on the state of the V54 bit (bit 7) of register AR10.

AR0-ARF Access = 0x3C1h/0x3C0h Regoffset =

7	6	5	4	3	2	1	0
Rsv		CV					
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-6	**Rsv**	**Reserved.** Must be written as zero.
Bits 5-0	**CV**	**6-bit Colour Value.**

11.7.3. ATTRIBUTE CTRL MODE REGISTER (RW)

AR10 Access = 0x3C1h/0x3C0h Regoffset =

7	6	5	4	3	2	1	0
V54	PW	PPC	Rsv	BE	LGE	MGE	GM
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**V54**	**V54 Select.** This bit determines whether bits 5-4 of the VGA pixel output come from the Video54 field of AR14 (bits 1-0) or from the normal output of the VGA Palette registers. Setting this bit to one selects the Video54 field.
Bit 6	**PW**	**Pixel Width.** When this bit is set to one, pixels are clocked at half the normal rate. The effect is to double the width of pixels displayed on the CRT.
Bit 5	**PPC**	**Pixel Panning Compatibility.** When VGA split screen is in effect, this bit controls whether both screens or just the top one are affected by Pixel and Byte Panning fields. When set to zero, both screens pan together.
Bit 4	**Rsv**	**Reserved.**
Bit 3	**BE**	**Blink Enable.** Setting this to one enables blinking in both text and graphics modes. When this bit is set to one in text mode, character attribute bit 7 is used on a character by character basis to enable or disable blinking. When this bit is set to zero in text mode, character attribute bit 7 controls character intensity. The blinking rate is equal to the vertical retrace rate divided by 32 (about twice per second). Setting this bit to one in graphics modes causes the VGA palette input bit 3 to toggle (approx twice per second) if the incoming pixel bit 3 is high.
Bit 2	**LGE**	**Line Graphics Enable.** Setting this bit to one forces the ninth pixel of a line graphics character (ascii codes C0h through DFh) to be the same as the eighth pixel. Setting it to zero forces the ninth pixel to be displayed as the background colour. Ninth pixels of all other ascii codes are always displayed as background colour. This bit has no meaning when character width is not set to nine or during graphics modes.
Bit 1	**MGE**	**Mono Attributes Enable.** Setting this bit to one in graphics modes while Bit 3 of this register is also one causes the VGA palette input bit 3 to toggle regardless of the incoming pixel's bit 3.
Bit 0	**GM**	**Graphics Mode.** Set this bit to one for graphics mode, zero for text mode.

11.7.4. ATTRIBUTE CTRL OVERSCAN COLOUR REGISTER (RW)

AR11 Access = 0x3C1h/0x3C0h Regoffset =

7	6	5	4	3	2	1	0
BC							
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-0	**BC**	**Border Colour.** These bits define the colour of the CRT border if there is one. The border or overscan region is that part of the display between where active pixels are displayed and those where the blank signal is active.

11.7.5. ATTRIBUTE COLOUR PLANE ENABLE REGISTER (RW)

AR12 Access = 0x3C1h/0x3C0h Regoffset =

7	6	5	4	3	2	1	0
Rsv		VSMC		CPE			
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-6	**Rsv**	**Reserved.** These bits should be written as zero.
Bits 5-4	**VSMC**	**Video Status Mux Control.** These bits select two of the eight output bits of the Attribute Controller to be read via Input Status Register #1 (3XAh) bits 5-4. The selection is as given inTable 11-12.
Bits 3-0	**CPE**	**Colour Plane Enable.** These four bits are ANDed with the frame buffer data before being input into the Palette. If any of these bits are zero, the corresponding plane from the frame buffer will be masked out of the Palette look up.

Table 11-12. Video Status Mux Control

AR12[5]	AR12[4]	ISR[5]	ISR[4]
0	0	PD[2]	PD[0]
0	1	PD[5]	PD[4]
1	0	PD[3]	PD[1]
1	1	PD[7]	PD[6]

11.7.6. ATTRIBUTE HORZ PIXEL PANNING REGISTER (RW)

AR13 Access = 0x3C1h/0x3C0h Regoffset =

7	6	5	4	3	2	1	0
Rsv				HPP			
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-4	**Rsv**	**Reserved.** These bits should be written as zero.
Bits 3-0	**HPP**	**Horizontal Pixel Panning.** These bits specify the number of pixels by which to shift the display left (seeTable 11-13).

Table 11-13. Horizontal Pixel Panning

H Pixel Pan	Shift		
	9 pixels/chr	**8 pixels/chr**	**mode 13**
0	1 pixel left	0 pixels	0 pixels
1	2 pixels left	1 pixel left	0 pixels
2	3 pixels left	2 pixels left	1 pixel left
3	4 pixels left	3 pixels left	1 pixel left
4	5 pixels left	4 pixels left	2 pixels left
5	6 pixels left	5 pixels left	2 pixels left
6	7 pixels left	6 pixels left	3 pixels left
7	8 pixels left	7 pixels left	3 pixels left
8-15	0 pixels	undefined	undefined

11.7.7. ATTRIBUT COLOUR SELECT REGISTER (RW)

AR14 Access = 0x3C1h/0x3C0h Regoffset =

7	6	5	4	3	2	1	0
Rsv				V76		V54	
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-4	**Rsv**	**Reserved.** These bits should be written as zero.
Bits 3-2	**V76**	**Video76.** In all modes except 256 colour mode (mode 13), these bits are output onto bits 7-6 of the VGA pixel data output port.
Bits 1-0	**V54**	**Video54.** When bit 7 of AR10 is set to '1', these bits are output onto bits 5-4 of the VGA pixel data output port.

11.8. CRT CONTROLLER REGISTERS

The STPC implements an extension of the VGA CRTC controller. The CRTC controller supports up to 1024 x 768 display resolutions at 75 HZ refresh rates as defined by VESA Monitor Timing Standard. The horizontal timing control fields are all VGA compatible.

The vertical timings are extended by 1-bit to accommodate above display resolution. The address registers are extended to allow locating the frame buffer in anywhere within the first 4 Mb of physical main memory.

11.8.1. INDEX REGISTER (RW)

The CRTC Index register points to an internal register of the CRT controller. The seven least significant bits determine which register will be pointed to in the next register read/write operation to IO port 3B5/3D5.

CRX Access = 0x3X4h/0x3X5h Regoffset =

7	6	5	4	3	2	1	0
Rsv	CRTC I						
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved.** Must be written to ‘0’s. Read back is undefined.
Bits 6-0	**CRTC I**	**CRTC Index.** Points to the CRTC register that will be accessed by an IO cycle at 03B5h/03D5h.

11.8.2. HORIZONTAL TOTAL REGISTER (RW)

The horizontal total register defines the total number of characters in a horizontal scan line, including the retrace time. The characters displayed on the screen are counted by a character counter. A count of 0 corresponds to the first displayed character at the left side of the screen. The value of the character counter is compared with the value in this register to provide the horizontal timing. A character is composed of 8 or 9 pixels as defined in Sequencer clocking mode register. All horizontal and vertical timing is based on the contents of this register.

The maximum horizontal resolution possible with this field is approximately 260*8*0.8 = 1664. (260 is 255+5, 0.8 is the fraction of a horizontal scan period during which active pixels are displayed).

CR0 Access = 0x3X4h/0x3X5h Regoffset = 000h

7	6	5	4	3	2	1	0
Default value after reset = 00h							

Programming notes:

The 8-bit value in this register = Total number of characters - 5.

11.8.3. HORIZ DISPLAY END REGISTER (RW)

This 8-bit read/write register defines the total number of displayed characters in a scan line.

CR1 Access = 0x3X4h/0x3X5h Regoffset = 001h

7	6	5	4	3	2	1	0
Default value after reset = undefined							

Programming notes:

The 8-bit value in this register = Total number of displayed characters - 1.

11.8.4. HORIZ BLANKING START REGISTER (RW)

This 8-bit read/write register defines when the horizontal blanking will start. The horizontal blanking signal becomes active when the horizontal character count equals the contents of this register.

CR2 Access = 0x3X4h/0x3X5h Regoffset = 002h

7	6	5	4	3	2	1	0
Default value after reset = undefined							

11.8.5. HORIZ BLANKING END REGISTER (RW)

CR3			Access = 0x3X4h/0x3X5h				Regoffset = 003h
7	6	5	4	3	2	1	0
Rsv	DESC		HBEV				
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved.** This readable and writable bit must be written to as ‘1’ to ensure proper VGA operation. It resets to one.
Bits 6-5	**DESC**	**Display Enable Skew Control.** These bits delay the display enable by the specified number of character clocks. The result is that the video output stream is delayed by the same amount resulting in wider left border and shrunk right border. This field is in unknown state after reset.
Bits 4-0	**HBEV**	**Horizontal Blanking End Value Bits 4-0.** These bits specify the least significant 5-bits of the 6-bit wide Horizontal Blanking End value. The sixth bit is located in CRTC Horizontal Retrace End register. This field is in unknown state after reset.

Programming notes:

This field controls the width of the horizontal blanking signal as follows:

Horizontal Blanking start register + width of the blanking signal = 6-bit Horizontal blanking end value.

The blanking signal set in CR2 and CR3 should start at least 23 GCLKs prior to the start of video window.

11.8.6. HORIZ RETRACE START REGISTER (RW)

This 8-bit register defines the character position at which the horizontal sync becomes active.

CR4 Access = 0x3X4h/0x3X5h Regoffset = 004h

7	6	5	4	3	2	1	0
Default value after reset = undefined							

11.8.7. HORIZONTAL RETRACE END REGISTER (RW)

CR5 Access = 0x3X4h/0x3X5h Regoffset = 005h

7	6	5	4	3	2	1	0
HBEV	HRSC		HRWV				
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bit 7	**HBEV**	**Horizontal Blanking End Value Bit 6.** This is the sixth bit of the Horizontal Blanking end field. Refer to CRTC Horizontal Blanking end register for more details.
Bits 6-5	**HRSC**	**Horizontal Retrace Skew Control.** This field delays the start of the horizontal sync by the specified number of character clocks. For text mode operation, this field should be programmed to ‘1’.
Bits 4-0	**HRWV**	**Horizontal Retrace Width Value.** These 5 bits specify the width of the horizontal sync signal as follows: Horizontal Retrace Start register + width of the horizontal sync = 5-bit Horizontal retrace end value.

11.8.8. VERTICAL TOTAL REGISTER (RW)

This register contains the least significant 8 bits of the 11-bit wide Vertical Total value. Next most significant 2 bits are located in CRTC overflow register CR7 and the 11th bit is located in Repaint Control Register 4.

CR6 Access = 0x3X4h/0x3X5h Regoffset = 006h

7	6	5	4	3	2	1	0
Default value after reset = undefined							

Programming notes:

The value programmed in this register = Total number of scan lines - 2.

11.8.9. OVERFLOW REGISTER (RW)

CR7 Access = 0x3X4h/0x3X5h Regoffset = 007h

7	6	5	4	3	2	1	0
VR	VD	VT	L	VB	VR	VD	VT
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bit 7	**VR**	**Bit 9 of the 11-bit wide Vertical Retrace start register.**
Bit 6	**VD**	**Bit 9 of the 11-bit wide Vertical Display end register.**
Bit 5	**VT**	**Bit 9 of the 11-bit wide Vertical Total register.**
Bit 4	**L**	**Bit 8 of the 11-bit wide Line compare register.**
Bit 3	**VB**	**Bit 8 of the 11-bit wide Vertical Blanking start register.**
Bit 2	**VR**	**Bit 8 of the 11-bit wide Vertical Retrace start register.**
Bit 1	**VD**	**Bit 8 of the 11-bit wide Vertical Display end register.**
Bit 0	**VT**	**Bit 8 of the 11-bit wide Vertical Total register.**

11.8.10. SCREEN A PRESET ROW SCAN REGISTER (RW)

CR8 Access = 0x3X4h/0x3X5h Regoffset = 008h

7	6	5	4	3	2	1	0
Rsv	DS		SS				
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bit 7	Rsv	**Reserved**. Must be written to a ‘0’. Read back is undefined.
Bits 6-5	DS	**Display Shift**. This field is added to the memory address generated during the display. As a result, the display shifts left by one, two or three Bytes. For both alphanumeric and graphics modes, this implies a left shift by 8, 16 or 24 pixels respectively. This field is encoded as given in Table 11-14. When the line compare condition becomes true and pixel panning compatibility bit (AR10 bit 5) is a ‘1’, the outputs of bits 5 and 6 are forced ‘0’ until the start of the next vertical sync pulse.
Bits 4-0	SS	**Smooth Scroll.** This field can be used to implement smooth vertical scrolling. It specifies the starting row scan count of the character cell after a vertical retrace (assuming the scan lines of a character row are numbered starting with 0). Smooth vertical scrolling can be implemented by setting this register to a value between 1 and the value in CR9. As a result, after a vertical retrace, the display will start from the scan line specified in this field instead of 0. This field is effective only for the top half of the screen (Screen A) if split screen mode is in effect. Each horizontal scan increments the horizontal row scan counter and is reset to 0 when it reaches the character cell height value programmed in CR9. If this field is programmed to a value larger than the character cell height, the row scan counter will count up to 1Fh before rolling over. A ‘0’ in this field means no scrolling.

Table 11-14. Byte Panning

Bit 6	Bit 5	Byte Panning
0	0	0 Byte (display shifts 0 pixels left)
0	1	1 Byte (display shifts 8 pixels left)
1	0	2 Bytes (display shifts 16 pixels left)
1	1	3 Bytes (display shifts 24 pixels left)

11.8.11. CHARACTER CELL HEIGHT REGISTER (RW)

CR9 Access = 0x3X4h/0x3X5h Regoffset = 009h

7	6	5	4	3	2	1	0
SC	LC	VB	SLPR				
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bit 7	**SC**	**Scan Double.** When set to a ‘1’, this bit allows a 200-line mode to be displayed on 400 display scan lines by dividing the row scan counter clock by 2 to duplicate each scan line. Thus all row scan address counter based timing (including character height and cursor and underline locations) double, as measured in scan lines, when scan doubling is enabled. Scan doubling only effects the way in which data is displayed; it does not effect display timing. If this bit is set without changing anything else, data currently displayed will appear twice as tall; horizontal and vertical sync, blanking etc., will remain the same.
Bit 6	**LC**	**Bit-9 of the 11 bit wide Line Compare field.**
Bit 5	**VB**	**Bit-9 of the 11 bit wide Vertical Blank Start field.**
Bits 4-0	**SLPR**	**Scan Lines Per Row.** This field specifies the number of scan lines per character row minus one.

11.8.12. CURSOR START REGISTER (RW)

CRA Access = 0x3X4h/0x3X5h Regoffset = 00Ah

7	6	5	4	3	2	1	0
Rsv		CDE	CSSL				
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-6	**Rsv**	**Reserved.** Must be written to ‘0’s. Read back is undefined.
Bit 5	**CDE**	**Cursor Display Enable.** When set to a ‘0’, this bit enables displaying the cursor. Cursor is displayed only in alphanumeric mode. In graphics mode, the cursor is always disabled and this bit has no effect.
Bits 4-0	**CSSL**	**Cursor Start Scan Line.** This field, in conjunction with the Cursor End scan line, defines the shape of the cursor. The hardware cursor is represented as a block of pixels occupying a character position. This field determines the first scan line within the character box that should be filled in (the first scan line is numbered as 0). If the cursor start and end scan line numbers are the same, one scan line wide cursor will be displayed. If starting scan line number is larger than the end, no cursor will be displayed. This is illustrated in Cursor Start and End Registers Figure 11-1.

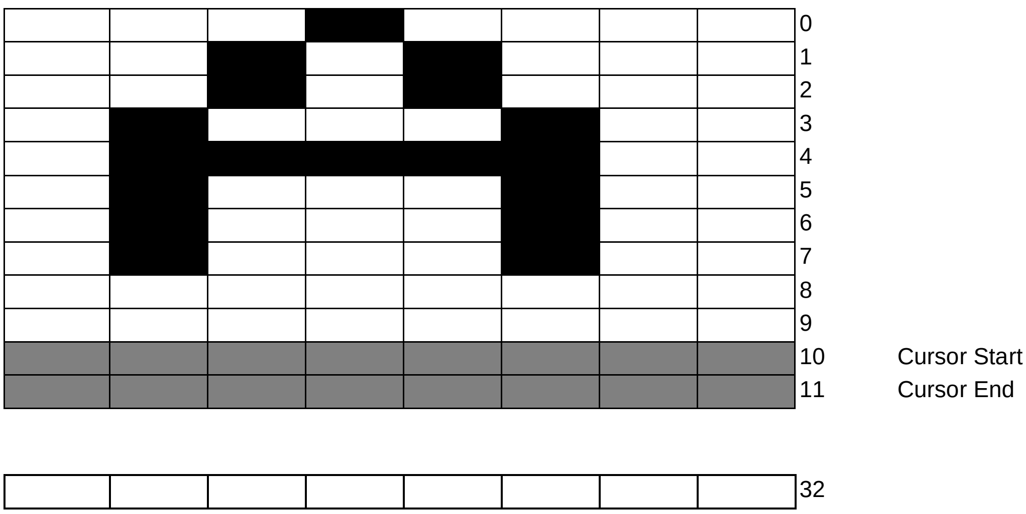


Figure 11-1. Cursor Start and End Registers

11.8.13. CURSOR END REGISTER (RW)

CRB Access = 0x3X4h/0x3X5h Regoffset = 00Bh

7	6	5	4	3	2	1	0
Rsv	CSC		CESL				
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved.** Must be written to ‘0’. Read back is undefined.
Bits 6-5	**CSC**	**Cursor Skew Control.** This field skews the cursor location (defined by the cursor location register) by the specified number of character clocks to the right.
Bit 5	**CESL**	**Cursor End Scan Line.** This field, in conjunction with the Cursor Start Scan line field defines the cursor shape. This is illustrated in Figure 11-1.

11.8.14. START ADDRESS HIGH REGISTER (RW)

This 8-bit register specifies bits 15-8 of the 20-bit display buffer address which will be displayed on the screen after a vertical retrace. Register CRD contains the lower 8-bits and the CRTC extended register CR19 contains the upper 4 bits.

If split screen mode is in effect, this address is the start address of the first of the two (the top one) screens (Screen A). The start address of Screen B (the bottom one) is always 0. The starting scan line for the Screen B is determined by the line compare register (CR18).

CRC Access = 0x3X4h/0x3X5h Regoffset = 00Ch

7	6	5	4	3	2	1	0
Default value after reset =							

11.8.15. START ADDRESS LOW REGISTER (RW)

This 8-bit register specifies bits 7-0 of the 20-bit display buffer address which will be displayed on the screen after a vertical retrace. Register CRC contains bits 15-8 and the CRTC extended register CR19 contains the upper 4 bits.

The start address in the CRC, CRD and CRIG does not define the offset within the frame buffer of the 1st displayed pixel but this value divided by 4.

CRD Access = 0x3X4h/0x3X5h Regoffset = 00Dh

7	6	5	4	3	2	1	0
Default value after reset =							

11.8.16. TEXT CURSOR OFFSET HIGH REGISTER (RW)

CRE Access = 0x3X4h/0x3X5h Regoffset = 00Eh

7	6	5	4	3	2	1	0
CO							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	**CO**	**Cursor offset bits 15-8.** This field contains the upper half of the 16-bit cursor offset. The offset is relative to the left-most character on the top of the screen and is specified in terms of character positions. For example, an offset of 0 will place the cursor on the left-most character on the top. An offset of 2 will place the cursor at the third character from the left in the top most row and so on. Since the information is stored in the display memory as character-attribute pairs, the address of the character under the cursor will be exactly twice the cursor offset + the screen base address.

11.8.17. TEXT CURSOR OFFSET LOW REGISTER (RW)

This register is the VGA compatible Cursor Offset Low register.

CRF Access = 0x3X4h/0x3X5h Regoffset = 00Fh

7	6	5	4	3	2	1	0
CO							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	**CO**	**Cursor offset bits 7-0.** This is the lower half of the cursor offset.

11.8.18. VERTICAL RETRACE START REGISTER (RW)

This register contains the lower 8 bits of the 11-bit wide vertical retrace start value. Register CR7 contains bits 8 and 9. Repaint Control Register 4 contains the msb of this 11-bit field. The retrace value is specified in horizontal scan lines where top most scan line on the screen is line 0.

CR10 Access = 0x3X4h/0x3X5h Regoffset = 010h

7	6	5	4	3	2	1	0
Default value after reset =							

11.8.19. VERTICAL RETRACE END REGISTER RW)

CR11			Access = 0x3X4h/0x3X5h				Regoffset = 011h
7	6	5	4	3	2	1	0
CR P	Rsv	VGA IE	VGA IR	VRW			
Default value after reset = 0x10xxxx							

Bit Number	Mnemonic	Description
Bit 7	**CR P**	**CR Protect.** This bit when set to a '1', write protects CR0-7 registers except CR7 bit 4.
Bit 6	**Rsv**	**Reserved.** This bit is both readable and writable.
Bit 5	**VGA IE**	**VGA Interrupt Enable.** When set to a '0', this bit enables the interrupt assertion of the VGA core. Setting this bit to a '1' disables the interrupts.
Bit 4	**VGA IR**	**VGA Interrupt Reset.** Setting this bit to a '0', clears the vertical retrace interrupt flip-flop and deasserts the interrupt output (if it was asserted). Setting this bit back to '1' enables the interrupt flip-flop to record the next vertical retrace. The interrupt flip-flop, if enabled by this bit, is set to one scan line after vertical blank is asserted. The flip-flop will not be set and the vertical retrace interrupt will be lost if this bit is set to a '0' when the interrupt occurred. The vertical interrupt flip-flop can be read as bit 7 of Input status register #0.
Bits 3-0	**VRW**	**Vertical Retrace Width.** These bits determines the width of the vertical retrace output as follows: Value in the Vertical Retrace Start register (CR10) + Width of the vertical retrace pulse = 4-bit value to be programmed into this field.

11.8.20. VERTICAL DISPLAY END REGISTER (RW)

This register contains the lower 8-bits of the 11-bit wide Vertical display end value which specifies the scan line position where the display on the screen ends. Bits 8 and 9 are specified in CRTC overflow register and the bit-10 in the Repaint Control Register 4.

CR12 Access = 0x3X4h/0x3X5h Regoffset = 012h

7	6	5	4	3	2	1	0
Default value after reset =							

Programming notes:

The value in this register = Total number of displayed scan lines - 1.

11.8.21. OFFSET REGISTER (RW)

This register defines bits 7-0 of the 10-bit wide logical width of the line displayed on the screen. Extended register CR19 contains the upper two bits. The first scan line displayed on the screen, starts at the address specified in registers CRC, CRD and extended register CR19. The starting address of the next scan line is computed as the Byte starting address of the current row + 2*offset.

CR13 Access = 0x3X4h/0x3X5h Regoffset = 013h

7	6	5	4	3	2	1	0
Default value after reset = undefined							

11.8.22. UNDERLINE LOCATION REGISTER (RW)

CR14 Access = 0x3X4h/0x3X5h Regoffset = 014h

7	6	5	4	3	2	1	0
Rsv	DWM	C 4	UL				UE
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved.** Must be written to a ‘0’. Read back is undefined.
Bit 6	**DWM**	**Double Word Mode.** If this bit is set to a ‘1’, the address generated by the CRTC memory address counter is shifted up two bits to provide the frame buffer address and bits 1-0 of the frame buffer address are driven from CRTC memory address counter bits 13 and 12 respectively. The logical screen width is multiplied by 8 and added to the starting address of the current scan line to compute the starting address of the next scan line.
Bit 5	**C 4**	**Count by 4.** Setting this bit to one causes the memory address counter to increment every four character clocks.
Bits 4-1	**UL**	**Underline Location.** This field specifies the horizontal row scan of the character cell at which the underlining will occur assuming that top line of the character cell is numbered 0. Underlining occurs in text (alphanumeric) modes only when an attribute value of ‘b000i001’ binary is detected (where b indicates blink and i indicates intensified).
Bit 0	**UE**	**Underline Enable.** Setting this bit to ‘1’ enables underlining.

Programming notes:

Underlining is enabled only in alphanumeric mode and then it is meaningful only for monochrome display (mode 7). For colour modes, the bit 0 of the attribute Byte is interpreted as foreground colour. There is no explicit bit to disable underlining for colour alphanumeric modes. Instead, it is disabled by programming this field to a value larger that the character cell height programmed in CR9.

11.8.23. VERTICAL BLANKING START REGISTER (RW)

This register contains the lower 8-bits of the 11-bit scan line valued where the vertical blanking is to begin. The 9th and 10th bits are located in CR7 and CR9 and the 11th bit is located in Repaint Control Register 4.

CR15 Access = 0x3X4h/0x3X5h Regoffset = 015h

7	6	5	4	3	2	1	0
Default value after reset = undefined							

11.8.24. VERTICAL BLANKING END REGISTER

This 8-bit register defines the width of the vertical blanking pulse as follows:

CR16 Access = 0x3X4h/0x3X5h Regoffset = 016h

7	6	5	4	3	2	1	0
Default value after reset =undefined							

Programming notes:

Start Vertical Blank value + width of the blanking pulse = Value programmed in this register.

While the register is 8-bits wide, and all bits are readable/writable, only the least significant 7 bits are used in the generation of the vertical pulse.

11.8.25. MODE REGISTER (RW)

CR17			Access = 0x3X4h/0x3X5h				Regoffset = 017h
7	6	5	4	3	2	1	0
H V RE	B WM	VGA MAS	Rsv	C 2	DVT	MS	MS
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**H V RE**	**H/V Retrace Enable.** A ‘0’ in this bit position disables the horizontal and vertical retraces and a ‘1’ enables them.
Bit 6	**B WM**	**Byte/Word# Mode**. A ‘1’ value in this bit position selects the Byte mode and a ‘0’ selects the word mode. Table 11-15 lists the memory address generation for Byte, word and double-word addressing modes. iA24-0 refer to the output of the internal memory address counter and the memory address is the address presented to the address lines of a 4-Byte wide display buffer memory, that is, each memory address selects 4-Bytes.
Bit 5	**VGA MAS**	**VGA Memory Address Size.** When set to a ‘0’, this bit, in Word addressing mode (see above) propagates bit iA13 on the least significant memory address bit. If set to a ‘1’, it propagates bit iA15 instead. iA13 should be used if total display buffer memory is 64K and iA15 should be used if total memory is 256K. It is expected that the VGA Controller will always be used with 256K or larger memory. Therefore this bit should be programmed to a ‘1’.
Bit 4	**Rsv**	***Reserved***. Must be written to a ‘0’. Read back is undefined.
Bit 3	**C 2**	**Count by 2.** Setting this bit to one causes the memory address counter to increment every second character clock.
Bit 2	**DVT**	**Double Vertical Total.** This bit when set to ‘1’, causes all the vertical timing counters to operate at half the horizontal retrace rate. The result is that the vertical resolution doubles. The Vertical Total, Vertical Retrace Start, Vertical Display End, Vertical Blanking Start and Line Compare registers can be programmed at half their normal value if this bit is set to a ‘1’. The vertical timing counters operate at their normal frequency if this bit is set to a ‘0’.
Bit 1	**MS**	**Memory Segmentation Bit 14.** If set to a ‘0’, the row scan counter bit 1 is substituted for memory address bit 14 during display refresh. This has the affect of segmenting the address space such that every other scan line pair is 8K apart. In combination with bit 0, this bit can segment the address space in 4-banks. No such substitution takes place if this bit is set to a ‘1’.
Bit 0	**MS**	**Memory Segmentation Bit 13.** This bit is similar to Bit 1 above in that when set to a ‘0’, row scan counter bit 0 is substituted for display memory address bit 13 during active display time. No such substitution takes place if this bit is set to a ‘1’.

Table 11-15. Memory Address Generation

Internal Memory Address counter	Byte Mode	Word Mode	Double Word Mode
iA24	iA24	iA23	iA22
iA23	iA23	iA22	iA21
:	:	:	:
:	:	:	:
iA3	iA3	iA2	iA1
iA2	iA2	iA1	iA0
iA1	iA1	iA0	iA13
iA0	iA0	iA13/iA15	iA12

The least significant memory address bit in Word mode is selected between iA13 and iA15 based on bit 5 of this register.

This bit is ignored if Double-word mode bit in CR14 is set to a ‘1’.

11.8.26. LINE COMPARE REGISTER (RW)

This register contains the lower 8 bits of the 10-bit wide Line Compare field. The 9th and 10th bits of this field are held in CR7 and CR9 registers respectively. When the horizontal scan line counter value is equal to the contents of the Line Compare register, the memory address generator and the character row scan count are cleared. As a result the display is split into the two halves. The top half, Screen A displays the contents of the display buffer starting from Start address (CRC and CRD registers) while the bottom half, Screen B, displays the contents of the display buffer starting from address 0.

CR18 Access = 0x3X4h/0x3X5h Regoffset = 018h

7	6	5	4	3	2	1	0
Default value after reset = undefined							

Programming Notes

Screen A can be smooth scrolled vertically but Screen B can not. Control is provided via bit 5 of AR10 register to allow Screen B to pan horizontally with Screen A or not.

Split screen function can be disabled by programming the Line compare field to a value larger, typically 3FFh, than the Vertical Total field. This field must be programmed to 3FFh for optimal system performance in native display modes.

11.8.27. GRAPHICS CONTROL DATA (R)

CR22 Access = 0x3X4h/0x3X5h Regoffset = 022h

7	6	5	4	3	2	1	0
GCDL N							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	**GCDL N**	**Graphics Controller Data Latch N.** These bits, when read, provide the state of one of the 4 Graphics Controller's Data Latches. The Graphics Controller Read Map Select register (GR4) specify which latch is read.

Programming notes:

The contents of this register are not altered by drawing operations.

11.8.28. ATTRIBUTE ADDRESS FLIP-FLOP (R)

CR24 Access = 0x3X4h/0x3X5h Regoffset = 024h

7	6	5	4	3	2	1	0
AF	Rsv						
Default value after reset =							

Bit Number	Mnemonic	Description
Bit 7	**AF**	**Attribute flip-flop.** This read-only bit indicates the state of the Attribute Controller index flip-flop. When this bit is zero, the next access to IO port 3C0h will be to the Attribute Index register. When this bit is one, the next access will be to an Attribute data register.
Bits 6-0	**Rsv**	**Reserved**. Read as zero.

11.8.29. ATTRIBUTE INDEX READBACK (R)

CR26 Access = 0x3X4h/0x3X5h Regoffset = 026h

7	6	5	4	3	2	1	0
Rsv		PAS	ACI				
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-6	Rsv	**Reserved.** Read as zero.
Bit 5	PAS	**Palette Address Source.** This is a read-only copy of Attribute Controller Index register (ARX) bit 5.
Bits 4-0	ACI	**Attribute Controller Index.** This is a read-only copy of bits 4-0 of the Attribute Controller Index register.

11.9. VGA EXTENDED REGISTERS

The following registers are additions to those found in the standard VGA specification. They can only be accessed after register SR6 has been written to with 57h.

A typical sequence in 80X86 assembly could be:

```
max     DX, 3C4h
mov     OX, 5706h
out     DX, AX
```

11.9.1. REPAINT CONTROL REGISTER 0 (RW)

CR19 Access = 0x3X4h/0x3X5h Regoffset = 019h

7	6	5	4	3	2	1	0
Rsv		CRTC O		CRTC SAF			
Default value after reset =00h							

Bit Number	Mnemonic	Description
Bits 7-6	**Rsv**	**Reserved**. Must be written to ‘0’.
Bits 5-4	**CRTC O**	**CRTC Offset register Bits 9-8.** See CRTC register 13 for details.
Bits 3-0	**CRTC SAF**	**CRTC Start Address Field Bits 19-16.** See CRTC register C, D for explanation of the Start Address.

11.9.2. REPAINT CONTROL REGISTER 1 (RW)

CR1A			Access = 0x3X4h/0x3X5h				Regoffset = 01Ah
7	6	5	4	3	2	1	0
HTD	VTD	Rsv	CTM	Rsv	LCE	SBP	VGA AW
Default value after reset = 3Fh							

Bit Number	Mnemonic	Description
Bit 7	**HTD**	**Hsync Toggle Disable.** When set to a ‘1’, this bit forces the Hsync to inactive state (high or low as programmed in the bit 6 of the Miscellaneous output register). See note below.
Bit 6	**VTD**	**Vsync Toggle Disable.** When set to a ‘1’, this bit forces the Vsync to inactive state (high or low as programmed in the bit 7 of the Miscellaneous output register). See note below.
Bit 5	**Rsv**	***Reserved***. This bit always reads as a one.
Bit 4	**CTM**	**Compatible Text Mode.** When this bit is set to one, the CRT controller expects the font data format within the frame buffer to be identical to that used by the standard VGA chip. Setting this bit to zero enables “Enhanced Text Mode” as described under CR1C bit 7. Note, though that this bit has the opposite sense of CR1C bit 7.
Bit 3	**Rsv**	**Reserved.** This bit always reads as a one.
Bit 2	**LCE**	**Line Compare Enable.** Set this bit to zero for 1280x1024 mode and one for all others.
Bit 1	**SBP**	**Six Bit Palette.** Set this bit to one to enable VGA compatible 6 bit palette functionality and zero to enable the 8-bit palette.
Bit 0	**VGA AW**	**VGA Address Wrap.** Setting this bit to one enables VGA compatible address wrapping such that the CRTC will only address 256kb of frame buffer memory (address bits 16 and above are zeroed). Set this bit to zero for SVGA modes. This bit has no effect on CPU reads or writes to the frame buffer - only CRTC accesses.

Programming notes:

Note: Vertical / Horizontal Synch not toggling is defined by the VESA specification for Monitor Power Down State.

11.9.3. REPAINT CONTROL REGISTER 2 (RW)

CR1B			Access = 0x3X4h/0x3X5h				Regoffset = 01Bh
7	6	5	4	3	2	1	0
FIFO LWM					Rsv	V FIFO U	W FIFO U
Default value after reset = 20h							

Bit Number	Mnemonic	Description
Bits 7-3	**FIFO LWM**	**FIFO Low Water Mark.** When the FIFO occupancy falls below twice this value, the CRTC will restart frame buffer read cycles to refill the FIFO. This field should only ever be set by the BIOS during mode switches. Setting this field too small results in random pixels being displayed to the screen; setting it too large results in decreased CPU - SDRAM bandwidth. Also, do not set this register to a value greater than that programmed into the high water mark (register CR27).
Bit 2	**Rsv**	**Reserved.**This bit is not writable. It reads as zeroes.
Bit 1	**V FIFO U**	**Video FIFO Underflow.** This read-only bit is set to one when the video refresh FIFO underflows. As with bit 0, writes to this register clear this bit to zero.
Bit 0	**W FIFO U**	**Warning: FIFO Underflow.** This read-only bit is set to one when the CRTC refresh FIFO underflows (the memory subsystem did not keep up with pixel requests. Sampling this bit as a one means that a serious problem exists and the low water mark above should be incremented. Writes to this register (presumably with a larger low water mark value) reset this bit to zero.

11.9.4. REPAINT CONTROL REGISTER 3 (RW)

CR1C			Access = 0x3X4h/0x3X5h				Regoffset = 01Ch
7	6	5	4	3	2	1	0
ETFL	Rsv		Rsv		SC	PSC	EOP
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**ETFL**	**Enhanced Text Font Load.** This bit should be set to one prior to loading fonts for 132 column high speed text mode. It warps the frame buffer addresses such that what appear to be standard text font writes actually get stored into frame buffer plane 2 in a more optimised manner. Specifically, frame buffer address bits 15-5 are swapped down to become bits 10-0 and bits 4-0 are moved up to become bits 15-11. The sequence of events to load 132 column enhanced text fonts is as follows. First, Odd-Even and Chain 4 modes should be turned off, then this bit should be set to one. Fonts should then be loaded in the normal VGA manner and finally this bit should be reset to zero and Text/Odd-Even mode entered. Note that the corresponding address warp for the CRT Controller is performed during font table look-ups when bit 4 of CR1A is set to zero.
Bits 6-5	**Rsv**	**Reserved,** read as ‘0’s.
Bits 4-3	**Rsv**	**Reserved.** These bits read as ‘0’s.
Bit 2	**SC**	**Sequential Chain-4.** When this bit is set to ‘1’, allows the display buffer memory to appear as a normal memory with a Byte address in the host address space mapping into a Byte address in the display buffer address space. Chain-4 in SR4 must be set for Sequential Chain-4 to work
Bit 1	**PSC**	**Page Select Control.** This bit provides control over whether the cycle type is read or write, or the upper address bit controls the page selection. The VGA implements two 7-bit page registers, Page 0 and Page 1, to allow mapping the VGA address space anywhere in the 4 MB address space. If this bit is ‘1’, the page selection is based on bit 15 or 16 of host address and the Memory Map bits of Register GR6 in Section 11.6.8. as given in Table 11-16.
Bit 0	**EOP**	**Enable Overlapped Paging.** This bit should be turned on to solve the broken line problem. When software wants to draw a line that crosses the current page boundary it turns this bit on to form a page out of half of the current page and half of the next page. Since the hardware adds half page to the address when this bit is on, the software should subtract half page for passing on the address. When this bit is ‘1’, the memory address bits MA17 and MA18 are changed as follows for Normal, Odd/Even and Chain-4 cases, see as given in Table 11-17. For Sequential Chain-4, MA16 and MA15 are changed as given in Table 11-18.

Table 11-16. Page Select

GR6 in Section 11.6.8.			Page Selection
Bit3	**Bit2**	**Size**	
0	0	128K	HA16=0 → Page 0; HA16=1 → Page 1
0	1	64K	HA15=0 → Page 0; HA15=1 → Page 1
1	0	32K	Not allowed
1	1	32K	Not allowed

Table 11-17. Enable Overlapped Paging

GR6 Section 11.6.8.			Added to MA18	Added to MA17
Bit3	**Bit2**	**Size**		
0	0	128K	1 (256K added)	0
0	1	64K	0	1 (128K added)
1	0	32K	Not allowed	
1	1	32K	Not allowed	

Table 11-18. Sequential Chain-4

GR6 Section 11.6.8.			Added to MA16	Added to MA15
Bit3	**Bit2**	**Size**		
0	0	128K	1 (64K added)	0
0	1	64K	0	1 (32K added)
1	0	32K	Not allowed	
1	1	32K	Not allowed	

Programming notes:

The contents of this register are not altered by drawing operations.

11.9.5. PAGE REGISTER 0 (RW)

CR1D Access = 0x3X4h/0x3X5h Regoffset = 01Dh

7	6	5	4	3	2	1	0
Rsv	P 0						
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved,** read as ‘0’.
Bits 6-0	**P 0**	**Page 0 Bits 6-0.** 7-bit Page register 0 is used to extend the host address to allow the VGA buffer to be located anywhere in the 4 MB frame buffer space. The pages are located on 32K boundaries for Normal, Odd/Even and Chain-4 modes and on 8K boundaries for Sequential Chain-4 mode. This is illustrated in Figure 11-2.

Programming notes:

Register = A000h mapped to the frame buffer and can be either 8 KBytes or 32 KBytes.

The contents of this register are not altered by drawing operations.

11.9.6. PAGE REGISTER 1 (RW)

CR1E			Access = 0x3X4h/0x3X5h				Regoffset = 01Eh
7	6	5	4	3	2	1	0
Rsv	P 1						
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved,** read as ‘0’.
Bits 6-0	**P 1**	**Page 1 Bits 6-0**. 7-bit Page register 1 is used to extend the host address to allow the VGA to be located anywhere in the 4 MB frame buffer space. The pages are located on 32K boundaries for Normal, Odd/Even and Chain-4 modes and on 8K boundaries for Sequential Chain-4 mode. This is illustrated in Figure 11-2.

Programming notes:

Register = A800h mapped to the frame buffer and can be either 8 KBytes or 32 KBytes.

The contents of this register are not altered by drawing operations.

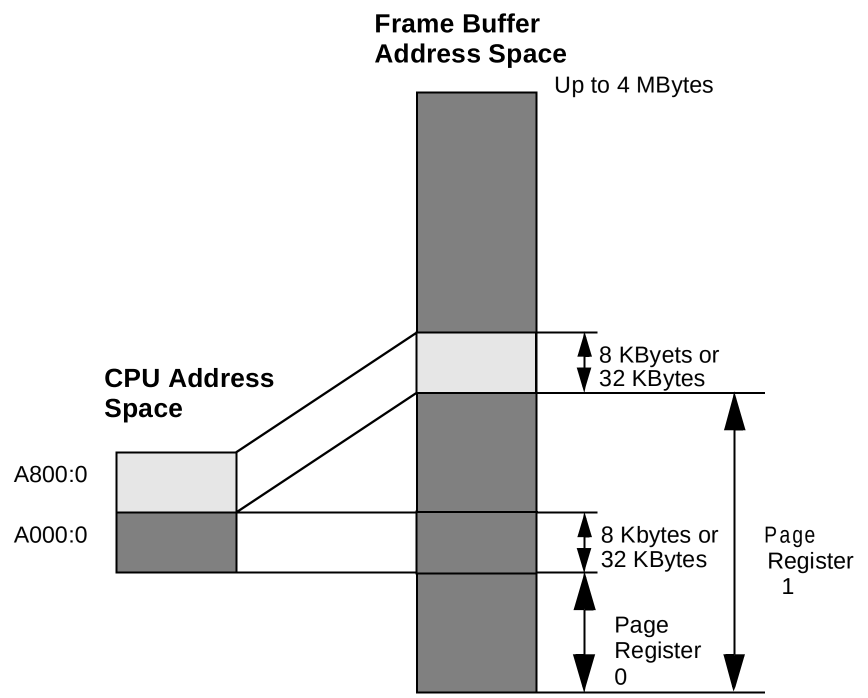


Figure 11-2. Illustration of Page Register 0 and Page Register 1

11.9.7. GRAPHICS EXTENDED ENABLE REGISTER (RW)

CR1F Access = 0x3X4h/0x3X5h Regoffset = 01Fh

7	6	5	4	3	2	1	0
E	Rsv						
Default value after reset =00h							

Bit Number	Mnemonic	Description
Bit 7	**E**	**Exten.** Writing a ‘1’ in this bit enables the GE extended functionality and also direct access to the frame buffer as defined by GBASE (CR20). Writing a ‘0’ disables it. After reset, this bit is set to ‘0’.
Bits 6-0	**Rsv**	**Reserved,** read as ‘0’.

Programming notes:

The contents of this register are not altered by drawing operations.

11.9.8. GRAPHICS EXTENDED GBASE REGISTER (RW)

CR20 Access = 0x3X4h/0x3X5h Regoffset = 020h

7	6	5	4	3	2	1	0
Rsv					G		
Default value after reset =00h							

Bit Number	Mnemonic	Description
Bits 7-3	**Rsv**	**Reserved,** these bits read as ‘0’.
Bits 2-0	**G**	**Gbase.** This range defines the bits 26 to 24 of the CPU address space where the GE Extended Frame Buffer and registers are located.

Programming notes:

The contents of this register are not altered by drawing operations.

11.9.9. GRAPHICS EXTENDED APERTURE REGISTER (RW)

CR21 Access = 0x3X4h/0x3X5h Regoffset = 021h

7	6	5	4	3	2	1	0
A							
Default value after reset =FFh							

Bit Number	Mnemonic	Description
Bits 7-0	**A**	**Aperture.** The lower 6 address bits are appended to the 16 least significant address bits in A0000h addresses to form a 22-bit address. This address is then used to map into the 4 MBytes Extended GE Register space. To use this feature, bits 7-6 must be set to ‘01’. Setting this register to FFh disables the aperture. Other values of this register can cause undefined results. The purpose of the aperture is to enable access to the extended memory mapped register in real mode.

Programming notes:

The contents of this register are not altered by drawing operations.

11.9.10. REPAINT CONTROL REGISTER 4 (RW)

CR25 Access = 0x3X4h/0x3X5h Regoffset = 025h

7	6	5	4	3	2	1	0
Rsv		Rsv	HB	VB	VR	VD	VT
Default value after reset =FFh							

Bit Number	Mnemonic	Description
Bits 7-6	**Rsv**	**Reserved**, read as ‘0’s.
Bit 5	**Rsv**	**Reserved.**
Bit 4	**HB**	**Bit 6 of the 7-bit wide Horizontal Blanking End register.**
Bit 3	**VB**	**Bit 10 of the 11-bit wide Vertical Blanking start register.**
Bit 2	**VR**	**Bit 10 of the 11-bit wide Vertical Retrace start register.**
Bit 1	**VD**	**Bit 10 of the 11-bit wide Vertical Display end register.**
Bit 0	**VT**	**Bit 10 of the 11-bit wide Vertical Total register.**

11.9.11. REPAINT CONTROL REGISTER 5 (RW)

CR27			Access = 0x3X4h/0x3X5h				Regoffset = 027h
7	6	5	4	3	2	1	0
FIFO HWM					Rsv		
Default value after reset =D0h							

Bit Number	Mnemonic	Description
Bits 7-3	**FIFO HWM**	**FIFO High Water Mark**. When the FIFO occupancy rises above this value, the CRTC will stop filling the FIFO. This field should only ever be set by the BIOS during mode switches. Do not set this register to a value greater than the default value D0h - nothing will work.
Bits 2-0	**Rsv**	**Reserved**, read as ‘0’.

11.9.12. PALETTE CONTROL REGISTER (RW)

CR28 Access = 0x3X4h/0x3X5h Regoffset = 028h

7	6	5	4	3	2	1	0
DAC PD	DAC S	SPD	Rsv	LUT B	P F		
Default value after reset =08h							

Bit Number	Mnemonic	Description
Bit 7	**DAC PD**	**DAC Power Down.** Setting this bit to one turns off the digital to analog converters. This is useful for when a second graphics card is installed in the system and power needs to be saved by turning the motherboard graphics off.
Bit 6	**DAC S**	**DAC Setup.** This bit specifies the blanking pedestal. Zero indicates a blanking pedestal of 0 IRE, one indicates 7.5 IRE.
Bit 5	**SPD**	**Sense Power Down.** Setting this bit to one forces the DDC monitor sense circuits to power down.
Bit 4	**Rsv**	**Reserved**. Must be written to ‘1’.
Bit 3	**LUT B**	**LUT Bypass.** Setting this bit to one bypasses the RAMDAC look up table (LUT) and allows pixels to drive the DACs directly. When this bit is set to zero the look up table is used to compute final pixel colours. This provides palette functionality for 8 bit and other low colour modes and gamma correction (non-linearity compensation) for the 24 and 32 bit true colour modes.
Bits 2-0	**P F**	**Pixel Format.** These 3-bits specify the pixel colour depth and are encoded as given inTable 11-19.

Table 11-19. Pixel Format

Bit 2	Bit 1	Bit 0	Pixel Format
0	0	0	VGA standard 8 bit
0	0	1	8 bit colour (non-VGA)
0	1	0	15-bit (555) direct colour
0	1	1	16-bit (565) direct colour
1	0	0	24-bit (888) direct colour
1	0	1	32-bit (8888) direct colour
	1	X	Reserved

Programming notes:

For 15-bit and 16-bit pixels, the 5 or 6 bits per colour are shifted left by 3 or 2 bits and then presented to the 8-bit DACs or LUT address (depending on bit 3 above). The least significant bits are set to zero.

The following resolutions are supported at 75 Hz refresh rate for each of the above colour depths when a 64 bit bank of SDRAM is used for the frame buffer.

Pixel Format	Maximum Resolution
VGA (other than mode 13)	1024 x 768
VGA (mode 13)	640 x 480
8 bit (non-VGA)	1024 x 768
15 bit	1024 x 768
16 bit	1024 x 768
24 bit	800 x 600
32 bit	640 x 480

Interlaced monitors and timings are supported.

11.9.13. CURSOR HEIGHT REGISTER

For the description of this register, see section 12.11.1. on page 352.

Must be written to ‘0’ when not using a Hardware cursor.

11.9.14. CURSOR COLOUR 0 REGISTER A

For the description of this register, see section 12.11.2. on page 353.

Must be written to ‘0’ when not using a Hardware cursor.

11.9.15. CURSOR COLOUR 0 REGISTER B

For the description of this register, see section 12.11.3. on page 354.

Must be written to ‘0’ when not using a Hardware cursor.

11.9.16. CURSOR COLOUR 0 REGISTER C

For the description of this register, see section 12.11.4. on page 355.

Must be written to ‘0’ when not using a Hardware cursor.

11.9.17. CURSOR COLOUR 1 REGISTER A

For the description of this register, see section 12.11.5. on page 356.

Must be written to ‘0’ when not using a Hardware cursor.

11.9.18. CURSOR COLOUR 1 REGISTER B

For the description of this register, see section 12.11.6. on page 357.

Must be written to ‘0’ when not using a Hardware cursor.

11.9.19. CURSOR COLOUR 1 REGISTER C

For the description of this register, see section 12.11.7. on page 358.

Must be written to ‘0’ when not using a Hardware cursor.

11.9.20. GRAPHICS CURSOR ADDRESS REGISTER 0

For the description of this register, see section 12.11.8. on page 359.

Must be written to ‘0’ when not using a Hardware cursor.

11.9.21. GRAPHICS CURSOR ADDRESS REGISTER 1

For the description of this register, see section 12.11.9. on page 360.

Must be written to ‘0’ when not using a Hardware cursor.

11.9.22. GRAPHICS CURSOR ADDRESS REGISTER 2

For the description of this register, see section 12.11.10. on page 361.

Must be written to ‘0’ when not using a Hardware cursor.

11.9.23. URGENT START REGISTER (RW)

CR33 Access = 0x3X4h/0x3X5h Regoffset = 033h

7	6	5	4	3	2	1	0
USP							
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-0	**USP**	**Urgent Start Position.** These bits represent the horizontal character count value at which urgency information will start to be generated for CRTC fetch requests. Prior to this position and after display enable negates, any CRTC fetches performed will be generated at low priority - i.e.. any CPU or blit operation will take precedence over CRTC regardless of CRTC FIFO occupancy. Once the horizontal character counter reaches this value, if CRTC FIFO occupancy is still below its low water mark then urgent fetches will be performed. Thereafter (and until the next display enable drops), as the CRTC FIFO drains, CRTC fetches will be marked urgent whenever the FIFO occupancy drops below its low water mark.

Programming notes:

A value of 0xFFh means “always urgent” and should be used if the VGA screen is showing “Glitches”. By setting this value, the CPU and GE bandwidth will be reduced.

11.9.24. DISPLAYED FRAME Y OFFSET 0 REGISTER (RW)

CR34 Access = 0x3X4h/0x3X5h Regoffset = 034h

7	6	5	4	3	2	1	0
F Y O							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	**F Y O**	**Frame Y Offset 0 Bits 7-0.** These bits represent bits 7-0 of a (2's complement) 16 bit scan line offset of the displayed frame relative to the Graphics Engine destination base.

11.9.25. DISPLAYED FRAME Y OFFSET 1 REGISTER (RW)

CR35 Access = 0x3X4h/0x3X5h Regoffset = 035h

7	6	5	4	3	2	1	0
F Y O							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	**F Y O**	**Frame Y Offset 1 Bits 7-0.** These bits represent bits 15-8 of the displayed frame scan line offset relative to the Graphics Engine destination base.

11.9.26. INTERLACE HALF FIELD START REGISTER (RW)

CR39 Access = 0x3X4h/0x3X5h Regoffset = 039h

7	6	5	4	3	2	1	0
I H C							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	**I H C**	**Interlace Horizontal Count.** This register defines the horizontal character count at which vertical timing is clocked during odd frames. When using interlaced operation, this register should be programmed to approximately half of the horizontal total value (CR0). There is no explicit interlace enable bit. Rather, when this register is programmed to FFh, interlace is disabled. The value of this register is defined to be FFh after reset (interlace disabled).

11.9.27. IMPLEMENTATION NUMBER REGISTER (R)

CR3A Access = 0x3X4h/0x3X5h Regoffset = 03Ah

7	6	5	4	3	2	1	0
I N							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	**I N**	**Implementation Number.** Indicates the hardware implementation number for the graphics drawing and display subsystem. Table 11-20 describes the interpretation of each value.

.

Table 11-20. Implementation Number

Value	Implementation
01h	STPC Implementation

11.9.28. GRAPHICS VERSION REGISTER (R)

CR3B			Access = 0x3X4h/0x3X5h				Regoffset = 03Bh
7	6	5	4	3	2	1	0
GVN							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	**GVN**	**Graphics Version Number.** Indicates the hardware version number for the graphics drawing and display subsystem. Table 11-21 describes the interpretation of each value.

.

Table 11-21. Graphics Version Number

Value	Implementation
01h	STPC Implementation

11.9.29. MISCELLANEOUS TEST REGISTER

CR3E			Access = 0x3X4h/0x3X5h				Regoffset = 03Eh
7	6	5	4	3	2	1	0
Rsv							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	**Rsv**	**Reserved**. Must be set to '0'.

11.9.30. DDC CONTROL REGISTER (RW)

CR3F Access = 0x3X4h/0x3X5h Regoffset = 03Fh

7	6	5	4	3	2	1	0
DDC WD		DDC RD		Rsv	Rsv		Rsv
Default value after reset = FFh							

Bit Number	Mnemonic	Description
Bits 7-6	**DDC WD**	**DDC Write Data.** These two bits drive the DDC[1:0] *Direct Data Channel Serial Link* pins. These bidirectional pins are connected to CRTC register 3Fh to implement DDC capabilities. They conform to I^2C electrical specifications and have open-collector output drivers which are internally connected to V_{DD} through pull-up resistors. Writes to these bits affect the DDC[1:0] pins. Thus, programming either of these bits to a one disables the output driver and allows the pin to act as an input whose status can be read via bits 5-4 of this register. They can instead be used for accessing I^2C devices on board. DDC1 and DDC0 correspond to SCL and SDA respectively. Note that reads from these bits return the value of data last written to this register. This may not be the same as the data actually on the bus if another master is driving it. Bits 5-4 of this register accurately reflect the data on the bus no matter what is driving it.
Bits 5-4	**DDC RD**	**DDC Read Data.** These read-only bits return the read status of the DDC[1:0] pins.
Bits 3-1	**Rsv**	**Reserved.** These bits are both readable and writable and must be programmed to ones to ensure future compatibility.
Bit 0	**Rsv**	**Reserved.** This bit must be programmed to '1' for correct operation.

11.9.31. TV INTERFACE CONTROL REGISTER (RW)

CR40 Access = 0x3X4h/0x3X5h Regoffset = 040h

7	6	5	4	3	2	1	0
TV IE	CCIR 656 E	VE	BTOE	VTV_HSYNC	FFA		
Default value after reset = FFh							

Bit Number	Mnemonic	Description
Bit 7	**TV IE**	**TV Interface Enable.** This bit enables the TV interface when set high.
Bit 6	**CCIR 656 E**	**CCIR-656 Enable.** When set to one, this bit enables the generation of CCIR-656 compatible timing codes onto the output pixel stream.
Bit 5	**VE**	**Video Enable** (active low). This bit multiplexes the TV output port between the graphics pipeline (bit 5 = ‘1’) and the video input port (bit 5 = ‘0’).
Bit 4	**BTOE**	**Bottom/Top Output Enable.** This bit controls the direction of the VTV_BT signal. When set to a one, VTV_BT is an output and is driven by the TV interface’s timing generator. Note that External Timing Generator 1, bit 29 also controls the direction of this signal. The truth table is as follows (see Table 11-22) .
Bit 3	**VTV_HSYNC**	VTV_**HSYNC Output Enable.** This bit controls the direction of the VTV_HSYNC signal. When set to a one, VTV_HSYNC is an output and is driven by the TV interface’s timing generator. Note that Video Input Port register 2B, bit 28 also controls the direction of this signal. The truth table is as follows (see Table 11-23)
Bits 2-0	**FFA**	**Flicker Filter Algorithm.** These bits control the operation of the anti-flicker filter according to Table 11-24

Table 11-22. VTV_BT Signal Direction

CR40 Bit [4]	vtg_ext1 **Bit [29]**	**VTV_BT direction**
0	0	input
0	1	output from VIP t/g
1	X	output from TVO t/g

Table 11-23. VTV_HSYNC Signal Direction

CR40 Bit [3]	vtg_ext1**Bit [28]**	**VTV_HSYNC direction**
0	0	input
0	1	output from VIP t/g
1	X	output from TVO t/g

Table 11-24. Anti-Flicker Filter Operation

Bit 2	Bit 1	Bit 0	Function
X	0	0	Filter disabled
0	0	1	0:1:1 2 tap filter
0	1	0	1:2:1 3 tap filter
0	1	1	1:3:1 3 tap filter
1	0	1	0:1:1 2 tap filter -Y only
1	1	0	1:2:1 3 tap filter - Y only
1	1	1	1:3:1 3 tap filter - Y only

11.9.32. TV HORIZONTAL ACTIVE VIDEO START A REGISTER (RW)

CR41 Access = 0x3X4h/0x3X5h Regoffset = 041h

7	6	5	4	3	2	1	0
TV							
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-0	**TV**	These are bits [7:0] of the eleven bit wide horizontal active video start field. This field controls the positioning of the left hand side of the active **TV** display - to a one pixel granularity.

11.9.33. TV HORIZONTAL ACTIVE VIDEO START B REGISTER (RW)

CR42 Access = 0x3X4h/0x3X5h Regoffset = 042h

7	6	5	4	3	2	1	0
Rsv	Rsv		See Below	TV	WH		
Default value after reset =							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved.** For test and/or debug purposes, this bit may be used to reverse the order of chrominance and luminance bytes in the output stream. Normal operation requires this bit to be written as a zero.
Bits 6-5	**Rsv**	**Reserved.** These read/write bits should be written as zeroes to ensure future compatibility.
Bit 4	**Rsv**	**Reserved.** These read/write bits should be written as zeroes to ensure future compatibility.
Bit 3	**TV**	**TV** output mode. This bit defines the output format on the four analog DAC outputs of the TV encoder. This bit defaults to zero on reset (see Table 11-26).
Bits 2-0	**WH**	These represent bits 10:8 of the eleven bit **Wide Horizontal** active video start field. See 11.9.32. on page 290.

Table 11-25. **TV Horizontal Acive Video Start B Modes**

Mode	TVHA bit 4	TVIC bit 4/3	VIPETG bit 29/28	Operation
Input 601 mode	0	0	0	Vsync (BT) and Hsync come from the external pads (CCIR 601) and go to VIP and DENC (pass through)
Input 656 mode	0	0	1	Vsync (BT) and Hsync come from the VIP (CCIR 656) and go to the DENC (pass through). Pads are outputs (VIP)
TV output	0	1	X	Vsync (BT) and Hsync come from the TV output. VIP sync lines are busy, thus the VIP can not be configured in the same mode as the TV output (CCIR 656/CCIR 601 or reversed). Pads are outputs (TVO)
Pipeline 601	1	0	0	Same as Input 601 mode but the DENC sync come from the TV output (CCIR 656 possible). Pads are inputs.
Pipeline 656	1	0	1	Same as Input 656 mode but the DENC sync come form the TV output (CCIR 601 possible). Pads are outputs (VIP)
TV independent output	1	1	X	Vsync (BT) and Hsync from the TV output to the DENC are independent from the VIP but are output on the pads and input to the VIP (crossed mode CCIR 601/656 not possible).

Table 11-26. TV Output Mode

Bit 3	RED_TV	GREEN_TV	BLUE_TV	CVBS
0	**red**	**green**	**blue**	**composite**
1	**chrominance**	**luminance**	**composite**	**composite**

11.9.34. TV HORIZONTAL SYNC END A REGISTER (RW)

CR43 Access = 0x3X4h/0x3X5h Regoffset = 043h

7	6	5	4	3	2	1	0
WH							
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-0	**WH**	These are bits [7:0] of the eleven bit **Wide Horizontal** sync end field. This field represents the width, in pixels, of the (interlaced) HSYNC signal produced by the TV interface. Bit 3 of CR40 controls whether this interlaced hsync is actually output.

11.9.35. TV HORIZONTAL SYNC END B REGISTER (RW)

CR44 Access = 0x3X4h/0x3X5h Regoffset = 044h

7	6	5	4	3	2	1	0
Rsv	Rsv	Rsv			WH		
Default value after reset =							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved.** This bit should be programmed to be the same as CR10 bit 0. It is separately programmable purely for test and / or debug purposes.
Bit 6	**Rsv**	**Reserved.** This bit, when set to one, allows the luminance output of the colourspace converter (progressive scan) to be directly output onto the TV_YUV output bus. This is for test/debug use only and this bit should normally only ever be programmed to zero.
Bits 5-3	**Rsv**	**Reserved.** These read/write bits should be written as zeroes to ensure future compatibility.
Bits 2-0	**WH**	These represent bits 10:8 of the eleven bit **Wide Horizontal** sync end field.

11.10. ADDITIONAL MODES

11.10.1. FAST 132 CHARACTER WIDE TEXT MODE.

To meet the high bandwidth requirements of 132 column text mode, VGA Controller supports a special high speed text mode. For column widths of 96 characters and greater, bit 7 of extended register CR1C - the Repaint Control Register #3 must be set to one prior to loading the font tables into frame buffer plane two. Fonts may then be loaded in the standard VGA manner one Byte at a time at the end of which bit 7 of CR1C should be reset to zero.

Setting bit 7 of CR1C to one performs an address warping such that standard VGA font load cycles actually store fonts into plane two the following way:

Byte 0: Character Set 0, Font (ASCII) 0, Line 0

Byte 1: Character Set 0, Font (ASCII) 1, Line 0

...

Byte 255: Character Set 0, Font 255, Line 0

Byte 256: Character Set 1, Font (ASCII) 0, Line 0

...

Byte 511: Character Set 1, Font 255, Line 0

Byte 512: Character Set 2, Font (ASCII) 0, Line 0

...

Byte 2047: Character Set 7, Font 255, Line 0

Byte 2048: Character Set 0, Font (ASCII) 0, Line 1

Applications which load their own fonts independent of the motherboard BIOS will not be supported in 132 column modes because of the above requirements.

Note that the above organization of font data will ensure that 132 column mode bandwidth requirements are low enough to be satisfied by 64 bit wide SDRAM Frame Buffers only. If the frame buffer is 32 bits wide, then the primary and secondary character map selects (SR3) should only ever be programmed such that both of the primary and secondary fonts are in the range 0-3 or both are in the range 4-7. Failure to observe this requirement will result in a garbaged screen.

11.11. INTERLACED MONITOR SUPPORT

Section 4.7.6.26a describes the “interlace half field start” register field. Setting this field to a value other than FFh (the power on reset default) enables interlaced CRT timing generation.

In interlaced timing mode, the horizontal and vertical timing parameters (CR0-CR7, CR10-CR12, CR15, CR16) should be programmed to values equal to what they would otherwise take in non-interlaced modes with the following modifications:

- Horizontal period must be an even number of character clocks. This results in the requirement that CR0[0] must equal ‘1’.
- Interlace half field start (CR39) must be set equal to CR4 - (CR0 + 5)/2.
- Vertical period must be an odd number of scan lines. That is, CR6[0] must be set to 1.
- Vertical overscan period should be an even number of scan lines. That is the vertical blank start field must be odd (CR15[0] = ‘1’) and vertical blank end field must be even (CR16[0] = ‘0’). If this is not

observed the top and bottom lines of the border will be only half a scan line wide on alternate fields. If no border will be displayed then there is no restriction on vertical blank start and end.

All other registers should be programmed as they would for the same resolution and colour depth in non-interlaced mode.

11.12. RAMDAC REGISTERS

11.12.1. PALETTE PIXEL MASK REGISTER (RW)

This eight-bit mask register is ANDed with the pseudo-colour pixel before doing the palette look-up. This provides an alternate way of altering the displayed colours without changing the display memory or colour palette.

Pixel_Mask Access = 0x3C6h Regoffset =

7	6	5	4	3	2	1	0
Default value after reset = FFh							

11.12.2. PALETTE READ INDEX REGISTER (W)

This register contains the index value for the read access to the 256 entries of the colour palette. Each entry is 24-bits wide (8-bits each for R, G and B) and is read as sequence of three Bytes. After writing the index of the entry to be read, the actual contents of the selected palette entry are read by doing three consecutive Byte reads from the DAC Data port (3C9h) in sequence: 1) Red, 2) Green and 3) blue. This 3-Byte read sequence is aborted and a new one is started if either the Read or Write Index register is written before reading the third Byte.

After the third Byte of the sequence is read, the index register is automatically incremented to point to the next entry of the look-up table. If the index is FFh, it rolls over to 00h.

This register is a write only register. Reads from this address do not return the contents of the index register. The Palette state register contents are returned instead.

Read_Index Access = 0x3C7h Regoffset =

7	6	5	4	3	2	1	0
Default value after reset =							

11.12.3. PALETTE STATE REGISTER (R)

This is a read only register and contains the two least significant bits of the last IO writes to IO address 3C6h-3C9h.

Palette_State Access = 0x3C7h Regoffset =

7	6	5	4	3	2	1	0
Rsv						IO	
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-2	**Rsv**	**Reserved.** The read back of this register is undefined.
Bit 1-0	**IO**	These bits contain the 2 LSBs of the address of the last IO write to ports 3C7h or 3C8h. ‘00’ indicate that the last write was to port 3C8h, ‘11’ indicate 3C7h.

11.12.4. PALETTE WRITE INDEX REGISTER (RW)

This register is similar to the Read index register and contains the index value for the write access to the 256 entries of the colour palette. Each entry is 24-bit wide and are written as a sequence of 3-Bytes. After writing the index of the entry to be modified, the data values may be written to the DAC Data port (3C9h) in the sequence: 1) Red, 2) Green, and 3) Blue. This 3-Byte write sequence is aborted and new one is started if either the Read or Write Index register is written before writing the third Byte.

After the third Byte of the sequence is written, the index register is automatically incremented to point to the next entry of the look-up table. If the index is FFh, it rolls over to 00h.

Both the Read and the Write index registers, physically map to a single index register. However only the Write Index register can be read. Reads from the Read Index register return the contents of Palette state register.

Write_Index Access = 0x3C8h Regoffset =

7	6	5	4	3	2	1	0
Not initialised by reset							

11.12.5. PALETTE DATA REGISTER (RW)

This register is used in conjunction with the Read and the Write index register to access the look-up table. Reads from this port return the contents of the entry pointed to by the Read Index register and writes to this port modify the content of the look-up table entry pointed to by the Write Index register. Each look-up table entry is 24-bit wide and is read or written as a sequence of three Bytes. The read or write sequence is always Red, Green and Blue. The normal procedure for accessing the look-up table is to initialise one of the Index registers and follow it with an uninterruptible sequence of three reads/writes from this register.

For VGA backward compatibility, when bits 2-0 of CR28 are programmed to 000 (as they are after reset), the palette look up table is treated as if each entry was only 18 bits wide. In this case, writes to port 3C9h map data such that bits 5-0 of host data are written into bits 7-2 of the look-up table while bits 1-0 are zeroed out. Similarly, reads return bits 7-2 of look-up table data onto host bits 5-0 and zero out bits 7-6.

When bit 5 of register CR3E is set to one, reads and writes to this port access red, green and blue signature data instead of look-up table data.

To minimise the sparkle while accessing the look-up table, all three bytes are read or written in a single video clock interrupting the screen repaint for one clock only. The interrupted pixel is painted with the same colour as the previous pixel.

Palette_Data Access = 0x3C9h Regoffset =

7	6	5	4	3	2	1	0
Not initialised by reset							

11.13 DCLK Control registers

These registers control the Dot Clock or pixel clock which the VGA uses to display the pixels on the screen.

11.13.1. DCLK Control Register 00

This is one of the 4-pairs of Dot Clock Control registers. This pair (00 and 01) is selected when bits 3-2 of VGA Miscellaneous Output register is set to 00.

DCLK00 Access = 022h/023h Regoffset =42h

7	6	5	4	3	2	1	0
Rsv	4BD				8BN		
Default value after reset = 0x76h							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved.**
Bits 6-3	**4BD**	This is the 4-bit M (divisor) value of the Dot Clock Synthesiser.
Bits 2-0	**8BN**	These are bits 7-5 of the 8 bit N (multiplier) of the Dot Clock synthesiser.

Programming notes:

This register defaults to 0x76h at reset. This value when combined with the default value of the other half of this pair results in a Dot Clock of 25.18 MHz assuming 14.318 MHz oscillator clock as the reference input.

$$DCLK = \frac{2 \times 14.31818 \times N}{M \times 2^P}$$

Constraints:

$$1 \leqslant M \leqslant 255$$
$$1 \leqslant N \leqslant 255$$
$$0 \leqslant P \leqslant 5$$

$$1\text{ MHz} \leqslant \frac{14.31818}{M} \leqslant 2\text{ MHz}$$

$$200\text{ MHz} \leqslant \frac{2 \times 14.31818 \times N}{M} \leqslant 622\text{ MHz}$$

11.13.2. DCLK control register 01

This is one of the 4-pairs of Dot Clock Control registers. This pair (00 and 01) is selected when bits 3-2 of VGA Miscellaneous Output register is set to 00.

DCLK01 Access = 022h/023h Regoffset =43h

7	6	5	4	3	2	1	0
3BP0	8BN					3BP1	
Default value after reset = 0x95h							

Bit Number	Mnemonic	Description
Bit 7	**3BP0**	This is the bit 0 of the 3-bit P (exponent) value of the Dot clock synthesiser.
Bits 6-2	**8BN**	These are bits 4-0 of the 8-bit N (multiplier) value of the Dot Clock Synthesiser.
Bits 1-0	**3BP1**	These are bits 2-1 of the 3-bit P (exponent) value of the Dot clock synthesiser.

Programming notes:

This register defaults to 0x95h at reset. This value when combined with the default value of the other half of this pair results in a Dot Clock of 25.18 MHz assuming 14.318 MHz oscillator clock as the reference input.

11.13.3. DCLK control register 10

This is one of the four pairs of dot clock control registers. This pair (10 and 11) is selected when bits 3-2 of VGA Miscellaneous Output register is set to 01.

DCLK10 Access = 022h/023h Regoffset =44h

7	6	5	4	3	2	1	0
Rsv	4BM				8BN		
Default value after reset = 0x76h							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved.**
Bits 6-3	**4BM**	This is the 4-bit M (divisor) value of the Dot Clock Synthesiser.
Bits 2-0	**8BN**	These are bits 7-5 of the 8 bit N (multiplier) of the Dot Clock Synthesiser.

Programming notes:

This register defaults to 0x76h at reset. This value when combined with the default value of the other half of this pair results in a Dot Clock of 28.0 MHz assuming 14.318 MHz oscillator clock as the reference input.

11.13.4. DCLK control register 11

This is one of the four pairs of Dot Clock Control registers. This pair (10 and 11) is selected when bits 3-2 of VGA Miscellaneous Output register is set to 01.

DCLK11 Access = 022h/023h Regoffset =45h

7	6	5	4	3	2	1	0
3BP0	8BN					3BP1	
Default value after reset = 0xEDh							

Bit Number	Mnemonic	Description
Bit 7	**3BP0**	This is the bit 0 of the 3-bit P (exponent) value of the Dot Clock Synthesiser.
Bits 6-2	**8BN**	These are bits 4-0 of the 8-bit N (multiplier) value of the Dot Clock Synthesiser.
Bits 1-0	**3BP1**	These are bits 2-1 of the 3-bit P (exponent) value of the Dot Clock Synthesiser.

Programming notes:

This register defaults to 0xEDh at reset. This value when combined with the default value of the other half of this pair results in a Dot Clock of 28.0 MHz assuming 14.318 MHz oscillator clock as the reference input.

11.13.5. DCLK control register 20

This is one of the four pairs of Dot Clock Control registers. This pair (20 and 21) is selected when bits 3-2 of VGA Miscellaneous Output register is set to 10.

DCLK20 Access = 022h/023h Regoffset =46h

7	6	5	4	3	2	1	0
Rsv	4BM				8BN		
Default value after reset = 0x5Bh							

Bit number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved.**
Bits 6-3	**4BM**	This is the 4-bit M (divisor) value of the Dot Clock Synthesiser.
Bits 2-0	**8BN**	These are bits 7-5 of the 8 bit N (multiplier) of the Dot Clock Synthesiser.

Programming notes

This register defaults to 0x5Bh at reset. This value when combined with the default value of the other half of this pair results in a dot clock of 40 MHz assuming 14.318 MHz oscillator clock as the reference input.

11.13.6. DCLK control register 21

This is one of the four pairs of Dot Clock Control registers. This pair (20 and 21) is selected when bits 3-2 of VGA Miscellaneous Output register is set to 10.

DCLK21 Access = 022h/023h Regoffset =47h

7	6	5	4	3	2	1	0
3BP0	8BN					3BP1	
Default value after reset = 0x6Dh							

Bit Number	Mnemonic	Description
Bit 7	**3BP0**	This is the bit 0 of the 3-bit P (exponent) value of the Dot Clock Synthesiser.
Bits 6-2	**8BN**	These are bits 4-0 of the 8-bit N (multiplier) value of the Dot Clock Synthesiser.
Bits 1-0	**3BP1**	These are bits 2-1 of the 3-bit P (exponent) value of the Dot Clock Synthesiser.

Programming notes:

This register defaults to 0x6Dh at reset. This value when combined with the default value of the other half of this pair results in a Dot Clock of 40 MHz assuming 14.318 MHz oscillator clock as the reference input.

11.13.7. DCLK control register 30

This is one of the four pairs of dot clock control registers. This pair (30 and 31) is selected when bits 3-2 of VGA Miscellaneous Output register is set to 11.

DCLK30 Access = 022h/023h Regoffset =48h

7	6	5	4	3	2	1	0
Rsv	4BM				8BN		
Default value after reset = 0x6Eh							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved.**
Bits 6-3	**4BM**	This is the 4-bit M (divisor) value of the Dot Clock Synthesiser.
Bits 2-0	**8BN**	These are bits 7-5 of the 8 bit N (multiplier) of the Dot Clock Synthesiser.

Programming notes:

This register defaults to 0x6Eh at reset. This value when combined with the default value of the other half of this pair results in a Dot Clock of 60 MHz assuming 14.318 MHz oscillator clock as the reference input.

11.13.8. DCLK control register 31 DCKL31 Index 49

This is one of the four pairs of dot clock control registers. This pair (30 and 31) is selected when bits 3-2 of VGA Miscellaneous Output register is set to 11.

DCLK31 Access = 022h/023h Regoffset =49h

7	6	5	4	3	2	1	0
3BP0	8BN					3BP1	
Default value after reset = 0x69h							

Bit Number	Mnemonic	Description
Bit 7	**3BP0**	This is the bit 0 of the 3-bit P (exponent) value of the Dot Clock Synthesiser.
Bits 6-2	**8BN**	These are bits 4-0 of the 8-bit N (multiplier) value of the Dot Clock Synthesiser.
Bits 1-0	**3BP1**	These are bits 2-1 of the 3-bit P (exponent) value of the Dot Clock Synthesiser.

Programming notes:

This register defaults to 0x69h at reset. This value when combined with the default value of the other half of this pair results in a Dot Clock of 60 MHz assuming 14.318 MHz oscillator clock as the reference input.

12. GRAPHICS ENGINE

12.1. INTRODUCTION

The Graphics Engine (GE) performs limited graphics drawing operations. The results of these operations change the content of the on-screen or off-screen frame buffer areas of DRAM memory.

Pixel depths of 8-bit, 16-bit, 24-bit and 32-bit are fully supported by the GE.

12.2. MEMORY ADDRESS SPACE

The extended (non-VGA) graphics and video functions of the Graphics Engine occupy 16 MBytes of memory address space. This space can be located anywhere in the memory on any 16 MByte boundary between 128 MBytes and 256 MBytes. The 16 MByte region is divided into four parts as shown in Figure 12-1.

Figure 12-1shows the GBASE as Extended CRTC Register 20 (CR20) and provides bits 26 through 24 of the starting address, where the CPU sees the extended graphics and video functionality.

This 16-MByte space can be linearly (one to one) mapped in the CPU address space or can be accessed via a 64K aperture located at A0000h-AFFFFh. The aperture access method (described in more detail in the VGA Controller Section of the Programming Manual, Graphics Extended Aperture Register, CR21) facilitates the access to extended functionality in real mode.

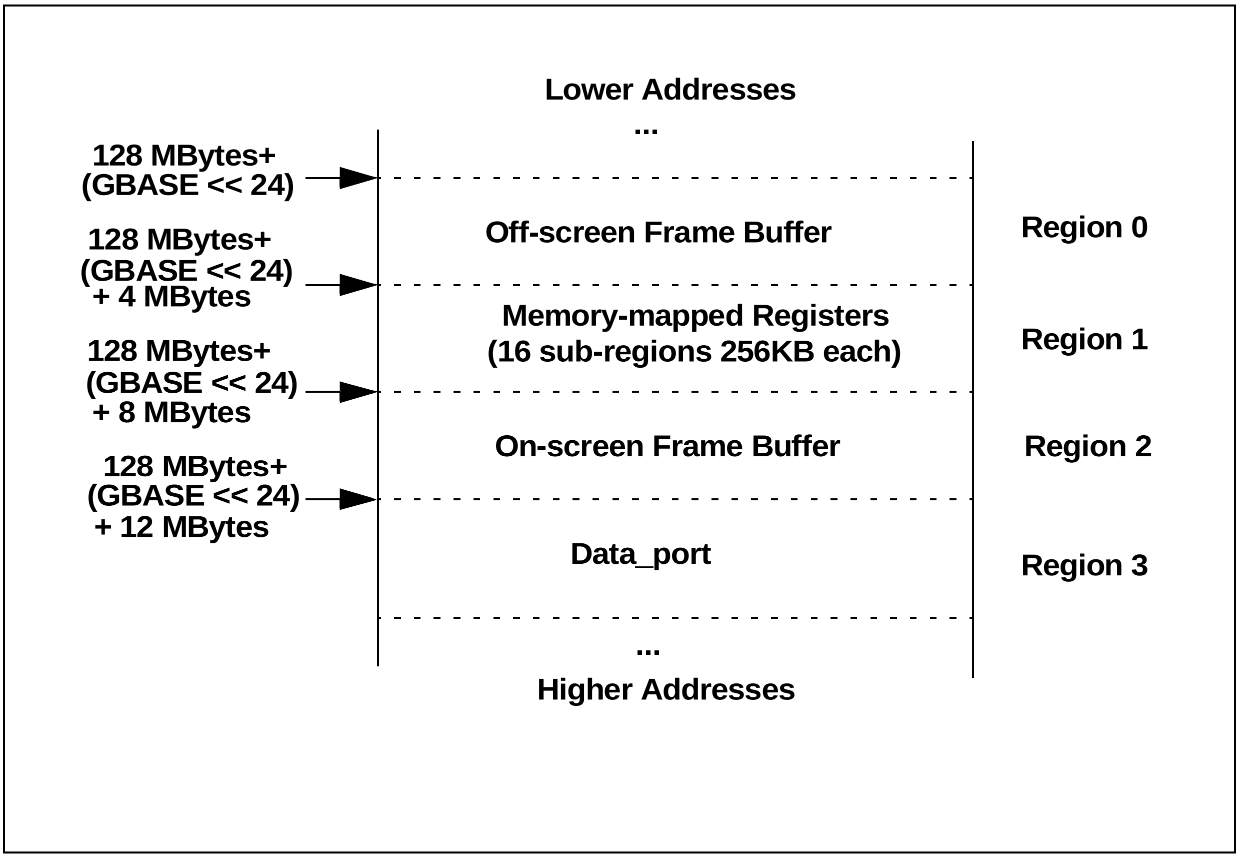


Figure 12-1. GE memory Map

Two 4-MByte regions are dedicated to the frame buffer, which reads from either 128 MBytes + (GBASE<<24) or 128 MBytes + (GBASE<<24) + 8 MBytes, handled identically in either case.

However, writes to any area of the Frame Buffer that might be displayed should be done to the region 128 MBytes + (GBASE<<24) + 8 MBytes. Writes to areas of the Frame Buffer that are not displayed should be done to 128MBytes + (GBASE<<24). The Frame Buffer address used by the host also informs the GE whether the access is being made to an on-screen or off-screen drawing area. It is important that the software never draws using an off-screen area address to a portion of the Frame Buffer that is currently being displayed. This is necessary to maintain compatibility with future versions of the GE.

The Frame Buffer addresses loaded into GE registers are from 0 to 4 MBytes. The source, pattern or destination of a GE operation can be located anywhere in Frame Buffer DRAM. However, the DRAM physical address must be known; the entire operand must be contiguous in physical memory (the GE does not include scatter/gather), and the operands must not move from the specified memory location until the drawing operation is completed.

All registers needed for the extended graphics and video functionality are mapped in a 4-MByte region of its own. This region is further divided in 16 X 256 KByte sub-regions illustrated in Table 12-1.

Table 12-1. Graphic Memory Sub-divisions

Sub-region	Region function
0	2-D Graphics Engine registers
1	Reserved
2	Video overlay registers
3-7	Reserved for future functionality
8	Video Input Port Registers
9-15	Reserved for future functionality

Writes done to any double word between (128 MBytes + (GBASE << 24) + 12 MBytes) to (128 MBytes + (GBASE << 24) + 16 MBytes) will be the same as a write to the Data_port register of the 2-D graphics engine. This region of 1 million aliases of the Data_port is provided to allow the use of string move instructions for Host-to-screen BitBlts.

All registers can be read with accesses of any width. The CPU can read any register via Byte (8-bit), word (16-bit), or double-word (32-bit) accesses. Writes must be done using double-word (32-bit) transfers.

Note that the contents of all GE registers are not defined after reset.

Software must initialise all registers upon power-up before attempting any drawing operation.

12.3. DUMB FRAME BUFFER ACCESS

The CPU can access the frame buffer memory as ordinary memory. It can read from or write to the frame buffer using any memory access instruction, and any data width. Thus, the CPU can access the frame buffer as if it were an unaccelerated display subsystem. This access by the CPU is permitted regardless of the GE busy status. Therefore, software must be careful to avoid race conditions or clashes if writing to the frame buffer when the GE is busy.

12.4. ADDRESSING

The GE frame buffer and extended registers may be accessed by the CPU via two methods: extended addresses, or A0000h-AFFFFh addresses. The former method allows direct access to the 16 MByte GE address range via 32-bit addresses. The A0000h-AFFFFh addressing method maps a 64K window of the 16 MByte GE address space into the address range A0000h-AFFFFh. For additional details on the A0000h-AFFFFh addressing, see the VGA Controller Section in the Programming Manual, Graphics Extended Aperture Register, CR21.

The Frame Buffer address used by the host also informs the GE whether the access is being made to an on-screen or off-screen drawing area. It is important that the software never draws using an off-screen area address to a portion of the Frame Buffer that is currently being displayed.

The addresses that are loaded into GE registers are the physical DRAM addresses after the OS address translation and GE host address remapping has been done. The Frame Buffer addresses loaded into GE registers are from 0 to 4 MBytes. The source, pattern or destination of a GE operation can be located anywhere in the Frame Buffer space.

12.5. VGA OPERAND SOURCES

The GE operates on data which can originate in one of three possible areas:

1) The frame buffer memory (i.e. a location in the DRAM memory that is dedicated to the graphics sub-system, and which may or may not be currently displayed by the CRT controller).

2) The host-supplied data.

3) The on-chip colour registers.

12.5.1. OPERAND SELECTION

Some operands are colour pixels and others are monochrome bitmaps. In general, the data written to the Destination address is the result of a Raster Operation (ROP) performed on three pixel-depth colour inputs:

1) The Source, which can originate from frame buffer memory (for Screen-to-screen BitBlts), from the Host (for Host-to-screen BitBlts), or from the Foreground and Background colour registers.

2) The pattern, which must originate from the frame buffer.

3) The destination, which must originate from the frame buffer.

When one or more of these operands are the inputs to an 8-bit Windows' ROP, the result is written to the destination.

If the ROP does not use a source operand, then the “Source” field of the ROP register must be set to CONSTANT_FILL (see ROP register Section 12.9.11.) to prevent wasted performance due to needless operand fetching.

If the ROP requires destination data reads, then the “Dst” field of the ROP register must be set to ‘1’. If destination reads are not required, then this field should be set to ‘0’.

If the ROP requires pattern data or uses colour transparent mode, then the “Pat” field of the ROP register must be set to ‘1’. If no pattern or colour transparency is being used in the operation, then this field should be set to ‘0’.

12.5.2. TRANSPARENT MODE

Transparent mode drawing leaves some of the destination pixels untouched. The GE supports four types of transparent mode drawing:

1) Bitmap transparency, where bits that are ‘1’ are expanded to the Foreground colour register value and drawn, but bits that are expanded to ‘0’ are not drawn.

2) Pattern transparency, where any Pattern Bytes that are ‘0’ suppress writing to the corresponding destination pixel Bytes.

3) Source transparency, where any Source pixel which either matches the value or does not match the value of the source transparency register is not drawn.

4) Destination transparency, where any Destination pixel that either matches the value or does not match the value of the source transparency register is not drawn.

These modes are controlled by fields in the ROP register.

12.6. VGA OPERAND FRAME BUFFER ADDRESSES

The GE fetches needed data from the frame buffer area. The software identifies these areas with an operand base address, unsigned X and Y Indices from this base address, and a pitch for that region. The pitch is the Byte distance between two pixels which are in the same X position of adjacent scan lines.

The frame buffer is addressed using DRAM linear addresses. These are the addresses that the DRAMs are presented with. The frame buffer starts at DRAM linear address 0 and continues until the top of the frame buffer. The system physical addresses are mapped to above the frame buffer. To accommodate a more natural view of the frame buffer, the GE implements X-Y addressing. An operand base address, pitch, X and Y components are combined in the GE, to form the associated DRAM linear address. The base component of an operand is the DRAM linear address at the start of that operand. Addresses can range from 0 to the maximum size of the frame buffer, depending on where the operand is located in the frame buffer.

A pixel X coordinate is usually expressed as an unsigned Byte quantity, the number of Bytes from the left edge of a scan line.

If the X_dir field of the Pixel_depth register is ‘0’, advancing from left-to-right, then X points to the least-significant Byte of the starting pixel.

If the X_dir field of the Pixel_depth register is ‘1’, advancing from right-to-left, then X points to the most-significant Byte of the starting pixel.

Mathematically, consider a BitBlt region that starts at (x0, y0), where “x0” is in pixels. This region is W+1 Bytes wide, is H+1 scan lines high and has BPP (Bytes-per-pixel).

Then the starting address that must be programmed into the GE is dependent on the X_dir and Y_dir fields of the Pixel_depth register. This is illustrated in Table 12-2.

Table 12-2. Detail GE Starting Address Register

X_dir	Y_dir	Starting Address
0	0	(x0 * BPP, y0)
0	1	(x0 * BPP, y0 + H)
1	0	(x0 * BPP + W, Y0)
1	1	(x0 * BPP + W, Y0 + H)

Note that movement in the negative X direction (i.e. X_dir set to ‘1’) is only defined for Screen-to-screen colour BitBlts.

When bitmap expansion is enabled, the X field of the Src_XY register is a bit address and not a Byte address. In other words, the least significant three bits of Src_XY.X refer to the bit within a Byte of bitmap data.

Internally, the GE performs calculations using the X and Y coordinates. When a DRAM linear address is needed, for example to write a destination pixel, the address is computed using:

linear_address = operand_base + (Y * pitch) + X *BPP

The multiplication by pitch is done using hardwired shifts and adds. The pitch is specified as a group of four shift codes. For each non-zero shift code, the Y address is shifted by a corresponding number of bits and then added to the total. The resulting sum is then added to X and the base address to obtain the DRAM linear address. The shift values supported are shown in Table 12-3.

Table 12-3. Shift Values Supported

Value	Shift0	Shift1	Shift2	Shift3
000	0	0	0	0
001	0	0	32 * Y	1024 * Y
010	0	64 * Y	64 * Y	2048 * Y
011	128 * Y	128 * Y	128 * Y	4096 * Y
100	256 * Y	256 * Y	256 * Y	n/a
101	512 * Y	512 * Y	512 * Y	n/a
110	1024 * Y	1024 * Y	0	n/a
111	2048 * Y	0	0	n/a

The operand base addresses must be aligned to 32 Bytes (that is, the five least significant bits of the address must be zeros).

The supported pitches (in Bytes) are:

0, 32, 64, 96, 128, 160, 192, 224, 256, 288, 320, 352, 384, 416, 448, 512, 544, 576, 608, 640, 672, 704, 768, 800, 832, 896, 1024, 1056, 1088, 1120, 1152, 1184, 1216, 1248, 1280, 1312, 1344, 1376, 1408, 1440, 1472, 1536, 1568, 1600, 1632, 1664, 1696, 1728, 1792, 1824, 1856, 1920, 2048, 2080, 2112, 2144, 2176, 2208, 2240, 2272, 2304, 2336, 2368, 2400, 2432, 2464, 2496, 2560, 2592, 2624, 2656, 2688, 2720, 2752, 2816, 2848, 2880, 2944, 3072, 3104, 3136, 3168, 3200, 3232, 3264, 3328, 3360, 3392, 3456, 3584, 3616, 3648, 3712, 3840, 4096, 4128, 4160, 4192, 4224, 4256, 4288, 4320, 4352, 4384, 4416, 4448, 4480, 4512, 4544, 4608, 4640, 4672, 4704, 4736, 4768, 4800, 4864, 4896, 4928, 4992, 5120, 5152, 5184, 5216, 5248, 5280, 5312, 5376, 5408, 5440, 5504, 5632, 5664, 5696, 5760, 5888, 6144, 6176, 6208, 6240, 6272, 6304, 6336, 6400, 6432, 6464, 6528, 6656, 6688, 6720, 6784, 6912, 7168, 7200, 7232, 7296, 7424, 7680.

12.6.1. COMMAND INITIATION

The destination coordinate register, Dst_XY, appears multiple times in the address space. Reading from any of these appearances, or aliases, is equivalent. Writing to most of these aliases also has the effect of initiating a drawing command. Which command is begun depends upon the address written to. There is also an address that just provides write access to the Destination register, with no other side-effects.

The operations encoded in the Dst_XY register are shown in Table 12-5. where “GBASE” is the contents of the Extended CRTC Register 20 (CR20). Bits 23 through 16 are ‘01000001’ to identify this as a Dst_XY register access. This is shown in Table 12-4.

Table 12-4. CMD Operations

CMD	Operation
00	Simple BitBlt, all registers must be set up before this command is issued (the Count field is ignored)
01	Width-specified BitBlt, the Count field of the address is used as the width for the operation, all other relevant registers (height, ROP, etc.) must be set up before this command is issued
10	Height-specified BitBlt, the Count field of the address is used as the height for the operation, all other relevant registers (width, ROP, etc.) must be set up before this command is issued
11	Write to Dst_XY without starting a BitBlt operation, (used for diagnostic applications)

When the CPU writes to the Dst_XY register using one of these operation aliases, the register write is completed and then the associated operation is begun. Thus, to perform an operation, the CPU writes to all but the Destination register. Then the last write is done to the Dst_XY register using one of the above aliases.

The ROP register also has fields that control the Source data (Screen, Host or Foreground colour register), enables/disables bitmap expansion, determines if the drawing is done in one of the transparent modes.

Screen-to-screen BitBlts are done as a BitBlt with the Source set as the Screen and bitmap expansion disabled.

Host-to-screen BitBlts are done as a BitBlt with the Source set as Host and the bitmap expansion disabled.

Rectangular fills are done as a BitBlt with the Source set to the Foreground colour register and the bitmap expansion disabled.

Text drawing using bit-packed font data provided by the Host is done as BitBlt with the Source set as Host and the bitmap expansion enabled. A transparent mode may also be specified.

Lines are not generally supported by the GE, but horizontal and vertical line segments can be quickly implemented as Width-specified BitBlts with the Height register set to 0 (to indicate a single pixel high BitBlt), or as Height-specified BitBlts with the Width register set to one less than the pixel depth.

Table 12-5. Encoded Dst_XY Registers

		4-MByte region 1 (memory mapped regs)	256KByte sub-region 0				
31 27	26 24	23 22	21 18	17 - 16	15 14	13 2	1 - 0
00001	<GBASE>	0 1	0000	0 - 1	<Cmd>	<Count>	1 - 0

12.7. DRAWING ENGINE REGISTERS

The software controls the graphic drawing by writing to the GE registers to set-up and initiate an operation. Any data that must be provided by the host is written to the Data_port. The Data_port can be referenced via the Data_port “register” or via the 4 MB window of aliases.

All registers can be read with accesses of any width. The CPU can read any register via Byte (8-bit), word (16-bit), or double-word (32-bit) accesses.

A register that is exclusively for the use of software, “Xtra”, is included in the GE but has no influence on any drawing operation or on the display.

12.8. REGISTER ACCESS

Except for the Dst_XY register, discussed in the previous section, the memory-mapped GE registers and the Data Port are accessed by reading or writing to an address of the form. This is illustrated in Table 12-6., where “Index” specifies the offset of the register to be accessed from the start of the GE memory-mapped register address space. The least significant two bits of Index will always be ‘00’. The following sections will list the “Index” value along with a description of each register. Reads may be done in any width, but writes must be done as 32-bit or 64-bit transfers.

In general, the CPU should not write to any of the GE registers when the GE is busy. If such an access is done, the CPU may be held for a long period of time, possibly for the duration of a large BitBlt. Reads of GE registers (except for Status) may return invalid data if the GE is busy.

The Dst_XY, Src_XY, Width and Height registers are double buffered and the CPU may write the next values to these registers while a prior operation is being performed. The last write to any double buffered register done before a write to Dst_XY will be the one used for the next operation. Any writes done to a GE register after a write is done to the Dst_XY while the GE is busy will hold the CPU until the first operation is completed and the pending register values are used for the second operation. During normal operation, the CPU writes to the Dst_XY register for text and line segments, reducing the hold period as much as possible.

The GE Status register may be accessed at any time.

12.8.1. DATA PORT ACCESS

The CPU writes Host data to the GE through the Data Port for Text and Host-to-screen BitBlt operations. The Data Port appears as one of the registers, as discussed in the previous section. Behind the Data Port, the Data FIFO buffers incoming data from the CPU. The Data Port is also repeatedly aliassed in the upper 4 MBytes of the GE 16 MByte address space.

In normal operation for text drawing (done as a bitmap expanded Host-to-screen BitBlts), the CPUwrites exact amount of data to the Data Port for the current character, then starts on the next text character by writing to the Dst_XY register and finally writes data for the next text character. The current operation reads from the Data Port FIFO until its needs are met. Then the next BitBlt operation reads its data from the Data Port FIFO. To ensure correct results, the software must write the correct number of 32-bit double words to the Data Port FIFO for all BitBlt operations.

The CPU can write to the Data Port at any time. If the GE is unable to accept any additional writes to the Data Port, the CPU is held off until the write command can be accepted. If a PCI master requests bus access when the CPU has been held off for a long period of time (128 clocks cycles), then the GE forces a CPU retry via the backoff (BOFF) mechanism.

Table 12-6. GE and Data_Port Access

		4-MByte region 1 (memory mapped regs)			
			256KByte sub-region 0		
31 27	26 24	23 22	21 18	17 12	11 0
00001	<GBASE>	0 - 1	0000	000000	<Index>

12.9. REGISTER SPECIFICATION

The GE registers are listed in alphabetical order and defined below.

12.9.1. BACKGROUND COLOUR REGISTER

This register contains the full-colour value(s) that a ‘0’ bit is expanded to. This expansion is done for BitBlt operations with bitmap expansion specified in the Expand field of the ROP register, and for operations with the Source field of the ROP register set to CONSTANT_FILL.

For 8-bit pixels, only bits 7-0 are significant. For 16-bit pixels, only bits 15-0 are significant. For 24-bit pixels, only bits 23-0 are significant.

Background   Access = GBASE+400000h   Regoffset = 0x004h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BC															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BC															
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	BC	**Background Colour.** This is the colour to be used as the background when expanding bitmap ‘0’ values or when using CONSTANT_FILL as the source operand.

Programming notes:

The content of this register is not altered by drawing operations.

12.9.2. CURSOR COORDINATE REGISTER

This register contains the address of the upper-left-hand corner of the cursor. To eliminate the cursor, its address should be set to a value large enough so that none of the cursor is on the displayed screen. Note that when set to (0,0), the entire cursor may be displayed on the upper-left hand corner of the display.

Cursor_XY　　Access = GBASE+400000h　　Regoffset = 0x11Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv			CYUL												
Default value after reset = undefined															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv			CXUL												
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bits 31-29	Rsv	Reserved.
Bits 28-16	CYUL	**CYUL**. The Y location of the upper-left-hand corner of the cursor.
Bits 15-13	Rsv	Reserved.
Bits 12-0	CXUL	**CXUL**. The X location of the upper-left-hand corner of the cursor.

Programming notes:

To suppress cursor display, enter one more than the number of display scan lines into the Y field.

The contents of this register remain unaltered throughout drawing and display operations.

12.9.3. TOP OF DATA FIFO REGISTER

This write-only register is the port through which the CPU provides Host data.

Data_Port Access = GBASE+400000h Regoffset = 0x804h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DP															
Default value after reset = undefined															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DP															
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bits 31-0	DP	Data_Port.

Programming notes:

The CPU can write to the Data Port at any time. If the GE is unable to accept any additional writes to the Data Port, the CPU will be held off until the write can be accepted.

Note that writing to this address is the same as writing to any double word between:

(128 MBytes+(GBASE << 24) + 12 MBytes) to (128 MBytes+(GBASE << 24) + 16 MBytes).

The Data FIFO is empty after reset.

12.9.4. DESTINATION OPERAND BASE ADDRESS REGISTER

This register specifies the starting DRAM linear address of the destination operand (aligned to a 32 Byte boundary).

Dst_Base Access = GBASE+400000h Regoffset = 0x018h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv											DOB				
Default value after reset = undefined															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DOB															
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bits 31-21	Rsv	Reserved.
Bits 20-0	DOB	**DstOp_Base.** Base DRAM linear address of the destination operand with 16 Byte alignement. Lower five Bytes are reserved and are set to '0'.

Programming notes:

The contents of this register are not altered by drawing operations.

12.9.5. DESTINATION PITCH REGISTER

This register specifies the number of Bytes needed to advance from a pixel in one scan line of the Destination to the corresponding pixel in the next scan line. This value is always positive. The Y_dirfield of the Pixel_depth register controls how the operation advances to the next scan line.

Only a limited number of pitches are supported. The supported ones (in Bytes) are listed in the Src_pitch description.

This register can be accessed via 32-bit or 16-bit transfers.

Dst_Pitch   Access = GBASE+400000h   Regoffset = 0x028h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = undefined															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv					DS3		DS2			DS1			DS0		
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bits 31-11	Rsv	**Reserved**.
Bits 10-9	DS3	**Dst_shift3.** These bits specify an amount to multiply Dst_XY.Y, this result along with the other shift results, is added to the Dst_base and Dst_XY.X to compute the DRAM linear address of the destination pixel. See Table 12-7. for the multiplication values that this field can specify.
Bits 8-6	DS2	**Dst_shift2**. See Dst_shift3, above.
Bits 5-3	DS1	**Dst_shift1**. See Dst_shift3, above.
Bits 2-0	DS0	**Dst_shift0.** See Dst_shift3, above.

Table 12-7. DRAM Address Multiplication Factor

Value	Shift0	Shift1	Shift2	Shift3
000	0	0	0	0
001	0	0	32 * Y	1024 * Y
010	0	64 * Y	64 * Y	2048 * Y
011	128 * Y	128 * Y	128 * Y	4096 * Y
100	256 * Y	256 * Y	256 * Y	n/a
101	512 * Y	512 * Y	512 * Y	n/a
110	1024 * Y	1024 * Y	0	n/a
111	2048 * Y	0	0	n/a

Programming notes:

The contents of this register are not altered by drawing operations.

12.9.6. DESTINATION OPERAND COORDINATE REGISTER

This register contains the coordinate address of the starting corner of the destination operand. The "starting" corner is controlled by the X_dir and Y_dst_dir fields of the Pixel_depth register.

Dst_XY Access = GBASE+400000h Regoffset =

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DY															
Default value after reset = undefined															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DX															
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bits 31-16	DY	**Dst_Y.** The unsigned Y coordinate of the starting corner of the destination operand.
Bits 15-0	DX	**Dst_X**. The unsigned X location of the starting corner of the destination operand. This value must be a multiple of Pixel_depth. The complete addressing of this register is described in Section 12.6.1. "Command Initiation".

Programming notes:

The address of this register is also used to determine which of the BitBlt operations is to be performed (Simple BitBlt, Width-specified BitBlt or Height-specified BitBlt). Writing to this register initiates a graphics operation.

This register is double-buffered. While the GE is busy executing one operation, a new value may be safely written to this register for the next operation. If the buffer is full, then the CPU will be held off. This feature is implemented specifically to accelerate the Text and Line Segment operations.

The contents of this register are not altered by drawing operations.

12.9.7. FOREGROUND COLOUR REGISTER

This register contains the full-colour value(s) that a “1” bit is expanded to.

Foreground Access = GBASE+400000h Regoffset = 0x034h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FCI															
Default value after reset = FFFFFFFFh															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FCI															
Default value after reset = FFFFFFFFh															

Bit Number	Mnemonic	Description
Bits 31-0	FCI	**Frg_Cl.** This is the colour to be used as the foreground when expanding bitmap ‘1’ values.

:

Programming notes:

This expansion is done for BitBlt operations with bitmap expansion specified in the Expand field of the ROP register.

For 8-bit pixels, only bits 7-0 are significant. For 16-bit pixels, only bits 15-0 are significant. For 24-bit pixels, only bits 23-0 are significant.

The contents of this register are not altered by drawing operations.

12.9.8. HEIGHT REGISTER

This register contains one less than the number of scan lines in the Source and Destination areas. The contents of this register will not change during the execution of a command.

Height | Access = GBASE+400000h | Regoffset = 0x048h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = undefined															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
H															
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bits 31-16	Rsv	Reserved.
Bits 15-0	H	**Height.** The value set in these bits must be one less than the height, in scan lines, of the source and destination areas.

Programming notes:

This register can be accessed via 32-bit or 16-bit transfers.

This register can be loaded by writing to the Dst_XY register using one the Height-specified address alias. For this case, the Height register is loaded with the value in the Count field of the address described in 12.6.1. "Command initiation".

This register is double-buffered. While the GE is busy executing one operation, a new value may be safely written to this register for the next operation. If the Dst_XY double-buffered register is full, then the CPU will be held off. This feature is specifically implemented to accelerate the Text and Line Segment operations.

The contents of this register are not altered by drawing operations.

12.9.9. PATTERN BASE ADDRESS OPERAND REGISTER

This register contains the starting DRAM linear address of the Pattern operand, including the aligned base address, the first row to be displayed and a starting Byte number for 24-bit pixels.

Pattern Access = GBASE+400000h Regoffset = 0x058h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv										PB					
Default value after reset = undefined															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB								Rsv			PXS		Rsv		
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bits 31-22	Rsv	Reserved.
Bits 21-8	PB	**Pattern base.** These bits specify the starting DRAM physical address of the Pattern, operand, aligned to a 256 Byte boundary.
Bits 7-5	Rsv	Reserved.
Bits 4-3	PXS	**Pat_X_start.** For 24-bit pixels, these bits must be set to: (Dst_X / 8) modulo 3 where Dst_X is the Byte address of the first 24-bit pixel in the destination row. For all other pixel depths, the values must be set to “00”.
Bits 2-0	Rsv	Reserved.

Programming notes:

The start of Pattern data must be aligned to a 256-Byte boundary. Advancing to the next Pattern data row will be done modulo 8 rows. Regardless of the number of Pixel_depth, the Pattern row is 32 Bytes long.

The Pattern register can be loaded with the address of the last row of Pattern data and the GE will wrap-around to the start of the pattern on the second row. Note that the Pattern register advances by increasing the address regardless of the X_dir, Y_src_dir or Y_dst_dir fields of the Pixel_depth register.

For further discussion of the Pattern Data, see Section 12.10.1. “Pattern Data”.

The contents of this register are not altered by drawing operations.

12.9.10. PIXEL DEPTH OPERAND REGISTER

This register contains the number of Bytes in a pixel, and bits that control the direction of Screen-to-screen BitBlts.

Pixel_Depth Access = GBASE+400000h Regoffset = 0x07Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = undefined															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv								Y	Y	X	Rsv			PD	
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bits 31-8	Rsv	Reserved.
Bit 7	Y	**Y_src_dir.** When this bit is set to ‘0’, source pixels advance from upper scan lines to lower scan lines (from smaller linear to larger linear addresses). Setting this bit to ‘1’ reverses the direction of BitBlt source operations. This bit should be set to ‘0’ for all operations other than reverse-direction non-bitmap-expanded screen-to-screen BitBlts.
Bit 6	Y	**Y_dst_dir.** When this bit is set to ‘0’ destination pixels advance from upper scan lines to lower scan lines (from smaller to larger linear addresses). Setting this field to ‘1’ reverses the direction of BitBlt destination operations. This field should be set to ‘0’ for all operations other than reverse-direction non-bitmap-expanded screen-to-screen BitBlts.
Bit 5	X	**X_dir.** When this bit is set to ‘0’, pixels advance from left to right, and when set to ‘1’ they advance from right to left. This field can be set to ‘1’ only for Screen-to-screen BitBlts and horizontal scan line fills.
Bits 4-2	Rsv	Reserved.
Bits 1-0	PD	**Pixel depth.** The only supported values for this field are shown in Table 12-8.

Table 12-8. Supported Pixel Depth Values

Bit 1	Bit 0	Pixel Depth
0	0	1 Byte per pixel
0	1	2 Bytes per pixel
1	0	3 Bytes per pixel
1	1	4 Bytes per pixel

Programming notes:

Note that the fourth Byte of 4-Byte pixels is not used in the display, but is processed by drawing operations. Zeros should be written to this 4th Byte to preserve compatibility with future versions of this architecture.

This register can be accessed via 32-bit or 16-bit transfers.

The contents of this register are not altered by drawing operations.

12.9.11. RASTER OPERATION REGISTER

This register contains the ROP code to be applied to process a pixel, to enable bitmap expansion, to select transparent modes, and to control the source operand. This is summarised in Table 12-9.

ROP Access = GBASE+400000h Regoffset = 0x08Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S		P	D	DM	Rsv										
Default value after reset = undefined															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D	D	S	S	P	E	P	B	R							
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bits 31-30	S	**Source.** These bits determine the SRC operand. Possible values are shown in Table 12-10. This field MUST be set to CONSTANT_FILL if the Raster Operation requires no SRC operand (such as in inverting the destination as well as constant fills). Failure to set this field correctly can result in a degradation of performance. Note that CPU writes to the Data Port will complete without error, and the data will be ignored unless the Source field is set to HOST.
Bit 29	P	**PAT present.** This bit is set to ‘1’ if a pattern data is to be used during the operation.
Bit 28	D	**DST present** This bit is set to ‘1’ if destination data is to be read during the operation.
Bit 27	DM	**Diagnostic Mode,** For normal operation this bit should be set to ‘0’. When set to ‘1’, GE register reads will be done from an alternative path for diagnostic verification.
Bits 26-16	Rsv	Reserved.
Bit 15	D	**DST transparency mode.** When this is set to ‘1’, pixels are selectively modified based upon a comparison of the DST data from the frame buffer vs. the DST transparency compare register. The results of the comparison are interpreted based upon the DST transparency match bit. Note that the DST present bit must also be set to ‘1’ when this bit is set. This mode is not valid when the pixel depth is 3 Bytes per pixel.
Bit 14	D	**DST transparency match.** This mode applies only when DST transparency mode is set. When this bit is set to ‘1’, pixels with DST data that match the DST transparency compare register will be modified. When it is set to ‘0’, pixels with DST data that do not match the DST transparency compare register will be modified.

Bit Number	Mnemonic	Description
Bit 13	S	**SRC transparency mode.** When this is set to ‘1’, pixels are selectively modified based upon a comparison of the SRC data vs. the SRC transparency compare register. The results of the comparison are interpreted based upon the SRC transparency match bit. This mode is only meaningful when using non-bitmap screen or host data as the source. Transparency for bitmap source data should not use this mode, but rather the SRC bitmap transparency mode. This mode is not valid when the pixel depth is 3 Bytes per pixel.
Bit 12	S	**SRC transparency match.** Applies only when SRC transparency mode is set. When this bit is set to ‘1’, pixels with SRC data that match the SRC transparency compare register will be modified. When this is set to ‘0’, pixels with SRC data that do not match the SRC transparency compare register will be modified.
Bit 11	P	**Packed.** If set to ‘1’, the source will be read in packed mode. Effectively, the source is viewed as a continuous stream of data. At the end of a destination scan line, any data remaining in the last-used source Dword is applied to the start of the next destination scan line. When this bit is set to ‘0’, any remaining source data is discarded at the end of a destination scan line. New source data is read from the next source scan line to apply to the start of the next destination scan line.
Bit 10	E	**Expand.** If set to ‘1’, the bitmap expansion will be enabled and source data from screen or host is assumed to be bitmap data. If set to ‘0’, source data is assumed to be colour data with the depth specified in the Pixel_depth register.
Bit 9	P	**PAT transparency mode.** When this bit is set to ‘1’, pixels are selectively modified based upon the value of corresponding pattern data. Pattern Bytes that are set to zero are not modified. Note that the PAT present bit must also be set to ‘1’ when this bit is set.
Bit 8	B	**Bitmap transparency mode.** When this bit is set to ‘1’, pixels are selectively modified based upon the pre-expanded bitmap value. Pixels with corresponding bitmap values of zero are not modified. Pixels with corresponding bitmap values of one are written with the foreground value. Note that the expand bit must also be set when using this mode.
Bits 7-0	R	**ROP**, the raster operation used when computing a pixel result value.

Table 12-9. Summary of ROP Functions

Bits	Function
31:30	SRC operand type
29	Use PAT operand
28	Use DST operand
27	GE Diagnostic mode
26:16	Unused/Reserved
15	DST transparency mode
14	DST transparency match
13	SRC transparency mode
12	SRC transparency match
11	Packed SRC data
10	SRC bitmap expansion
9	PAT transparency mode
8	SRC bitmap transparency
7:0	Raster operation code

Table 12-10. Detail of SRC Operand Functions

Bit 31	Bit 30	Function	Source
0	0	CONSTANT_FILL	Background colour register
0	1	SCREEN	screen or frame buffer
1	0	HOST	host CPU
1	1	Reserved	

Programming notes:

The contents of this register are not altered by drawing operations.

12.9.12. SOURCE BASE ADDRESS OPERAND REGISTER

This register specifies the starting DRAM linear address of the source operand (aligned to a 32-Byte boundary).

Src_Base   Access = GBASE+400000h   Regoffset = 0x098h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv										SB					
Default value after reset = undefined															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SB											Rsv				
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bits 31-22	Rsv	Reserved.
Bits 21-5	SB	**SrcOp_Base.** Base linear address of the source operand.
Bits 4-0	Rsv	**Reserved.** These bits are set to ‘0’.

Programming notes:

The contents of this register are not altered by drawing operations.

12.9.13. SOURCE PITCH OPERAND REGISTER

This register specifies the number of Bytes needed to advance from a pixel in one scan line of the source to the corresponding pixel in the next scan line. This value is always positive. The Y_src_dir field of the Pixel_depth register controls how the operation advances to the next scan line.

Only a limited number of pitches are supported. The supported pitches (in Bytes) are described in Section 12.6. "VGA operand frame buffer addresses".

Src_Pitch Access = GBASE+400000h Regoffset = 0x0ACh

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = undefined															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv					S3		S2			S1			S0		
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bits 31-11	Rsv	Reserved.
Bits 10-9	S3	**Src_shift3.** These bits specify an amount to multiply Src_XY.Y, this result along with the other shift results, is added to the Src_base and Src_XY.X to compute the DRAM linear address of the source pixel. See the Table 12-11. below for the multiplication values that this field can specify.
Bits 8-6	S2	**Src_shift2.** See Src_shift3, above.
Bits 5-3	S1	**Src_shift1.** See Src_shift3, above.
Bits 2-0	S0	**Src_shift0.** See Src_shift3, above.

Table 12-11. Scr_shift3 Multiplication Factors

Value	Shift0	Shift1	Shift2	Shift3
000	0	0	0	0
001	0	0	32 * Y	1024 * Y
010	0	64 * Y	64 * Y	2048 * Y
011	128 * Y	128 * Y	128 * Y	4096 * Y
100	256 * Y	256 * Y	256 * Y	n/a
101	512 * Y	512 * Y	512 * Y	n/a
110	1024 * Y	1024 * Y	0	n/a
111	2048 * Y	0	0	n/a

Programming notes:

This register can be accessed via 32-bit or 16-bit transfers. The contents of this register are not altered by drawing operations.

12.9.14. SOURCE COORDINATE REGISTER

This register contains the coordinate address of the starting corner of the source operand.

Src_XY Access = GBASE+400000h Regoffset = 0x0BCh

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SY															
Default value after reset = undefined															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SX															
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bits 31-16	SY	**Src_Y.** The source operand starting corner of the unsigned Y coordinate.
Bits 15-0	SX	**Src_X.** The source operand starting corner of the unsigned X location. When bitmap expansion is not enabled, this is a Byte address. When in bitmap expansion is enabled, this is a bit address.

Programming notes:

The “starting” corner is controlled by the X_dir and Y_src_dir fields of the Pixel_depth register.

This register is double-buffered. While the GE is busy executing one operation, a new value may be safely written to this register for the next operation. If the Dst_XY double-buffered register is full, then the CPU will be held off.

The Y field and all but the lower three bits (five when bitmap expansion is enabled) of the X field are ignored during Host-to-screen BitBlts.

The contents of this register are not altered by drawing operations.

12.9.15. STATUS REGISTER

Status | Access = GBASE+400000h | Regoffset = 0x908h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GB	PB	Rsv													
0		Default value after reset = undefined													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv															
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bit 31	GB	**GE_Busy.** This is a read only bit is set to ‘1’ when the GE is busy, GE register accesses that are done when this bit is set may result in the CPU being held for the duration of the current operation.
Bit 30	PB	**Pending Busy.** This read-only bit is set to ‘1’ when the Dst_XY pending register has data in it. If GE writes to that registers when this bit is set it may result in the CPU being held for the duration of the current operation. GE register reads always return data without holding the CPU, but the data returned from the read may not be valid. The Status register may be read at any time and the operation will return valid data. Note that Pending Busy implies Busy, that is the Pending Busy field can be set to ‘1’ only if the Busy field is also set to ‘1’.
Bits 29-0	Rsv	**Reserved.** These may read as one or zero.

12.9.16. WIDTH REGISTER

This register contains the one Byte width less than the destination operand.

This register can also be loaded by writing to the Destination register using one of the Text or Line Segment commands. For these cases, the Width register is loaded with the value in the Count field of the address described in Section 12.6.1., “Command Initiation”.

This register is double-buffered. While the GE is busy executing one operation, a new value may be safely written to this register’s Width field for the next operation. If the Dst_XY double buffered register is full, then the CPU will be held off.

Width Access = GBASE+400000h Regoffset = 0x0C8h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = undefined															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
W															
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bits 31-16	Rsv	Reserved.
Bits 15-0	W	**Width.** These bits should be set to one less than the number of Bytes across the destination area. This value should be a multiple of Pixel_depth, because only the number of Bytes specified in this field will be modified.

Programming notes:

This register can be accessed via 32-bit or 16-bit transfers.

The contents of this register are not altered by drawing operations.

12.9.17. EXTRA USE REGISTER

This register contains 32 bits of data that software can read from and write to.

This register has no effect on any drawing operation or display.

Xtra Access = GBASE+400000h Regoffset = 0x0D4h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D															
Default value after reset = undefined															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D															
Default value after reset = undefined															

Bit Number	Mnemonic	Description
Bits 31-0	D	Data for user software use.

Programming notes:

The contents of this register are not altered by drawing operations.

12.9.18. SRC TRANSPARACENCY COMPARE REGISTER

This 32-bit register contains the pixel value used for comparison in SRC transparency mode. For pixels depths of one Byte per pixel, the pixel value must be replicated in all four Bytes of this register. For pixel depths of two Bytes per pixel, the pixel value must be replicated in the upper and lower 16 bits of this register.

SRC_Transparency Access = GBASE+400000h Regoffset = 0xECh

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D								C							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
B								A							

1 byte per pixel	colour replicated in A, B, C and D
2 bytes per pixel	colour replicated across A, B and C, D
3 bytes per pixel 24-bit colour	colour replicated across A, B and C
4 bytes per pixel 32-bit colour	colour replicated across A, B, C and D

12.9.19. DST TRANSPARENCY COMPARE REGISTER

This 32-bit register contains the pixel value used for comparison in DST transparency mode. For pixels depths of one Byte per pixel, the pixel value must be replicated in all four Bytes of this register. For pixel depths of two Bytes per pixel, the pixel value must be replicated in the upper and lower 16 bits of this register.

DST_Transparency Access = GBASE+400000h Regoffset = 0xFCh

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D								C							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
B								A							

1 byte per pixel	colour replicated in A, B, C and D
2 bytes per pixel	colour replicated across A, B and C, D
3 bytes per pixel 24-bit colour	colour replicated across A, B and C
4 bytes per pixel 32-bit colour	colour replicated across A, B, C and D

12.9.20. NOTES ON: INTERACTIONS BETWEEN BLT OPERATIONS AND VGA FRAMEBUFFER ACCESSES

The GE performs two major classes of operations: BitBlts and standard VGA Framebuffer accesses. These two types of operations share resources in the hardware. This imposes certain requirements on driver software.

The state of all standard VGA registers is unchanged by BitBlt and extended register reads/writes with the exception of the CR22 data latch. The state of this register is undefined after a BitBlt.

The state of all extended registers is unchanged by VGA read/write operations.

Between a BitBlt operation and a VGA read/write operation, the software must ensure that no BitBlt operation is in progress by means of the Status register.

Before performing any VGA read/write operations, the software must ensure the Foreground register has the value FFFFFFFh and the background register has the value 00000000h. These are also the reset values of these registers.

Between a VGA write operation and a BitBlt operations, the software must ensure the VGA write pipeline is flushed by performing a VGA read operation.

12.10. GE OPERATIONS

12.10.1. PATTERN DATA

If the ROP register value specifies that pattern data is used in the computation of the destination results, then one row of the pattern data is read at the start of each scan line processed. This row of data is repeatedly applied to the result computation across scan line. The Pattern register points to the start of an 8-pixel-by-8-pixel colour area that is aligned to the destination. The GE does not perform any horizontal alignment to the pattern data.

When the pixel depth is three Bytes, the least significant three bits of the Pattern register must indicate which Byte starts the pattern row. This field should be set to:

(Dst_X / 8) modulo 3

where Dst_X is the Byte address of the first 24-bit pixel in the destination row. (The same value that is written to the X field of the Dst_XY register).

Bitmap patterns are not directly supported. To use a bitmap pattern, first allocate off-screen frame buffer memory for a colour version of the pattern. Then set up the GE to perform a Host-to-screen BitBlt with bitmap expansion into this allocated memory. The bitmap pattern is then written to the Data Port. The expanded pattern can now be used by pointing the Pattern register to the allocated memory.

12.10.2. BITMAT CONSIDERATIONS

Screen-to-screen and Host-to-screen operations can optionally expand single-bit-per-pixel bitmaps into colour pixels. Each ‘1’ bit is replaced by the contents of the Foreground colour register and each ‘0’ bit is replaced by the contents of the Background colour register.

Bitmaps from the frame buffer (during Screen-to-screen BitBlts) must be aligned on a quad-word (64-bit) boundary. Bitmaps from the Host can be aligned on a double-word (32-bit) boundary. Leading bits of the bitmaps may be skipped by setting the least significant bits of the X field of the Src_XY register to the number of bits in the Byte to be ignored. When in bitmap expansion mode, the X field of the Src_XY address can be thought of as a bit address instead of a Byte address. For Host-to-screen bitmap expanded BitBlts only the least significant five bits of the Src_XY.X register are significant. The first bit after those skipped will then be aligned to the first destination pixel.

For bitmap expansion, the X_dir must be ‘0’. The result for a bitmap expansion BitBlt with X_dir set to ‘1’ is not defined.

With the X_dir field of the Pixel_depth register set to ‘0’, the bitmap is considered to start at the least significant end of the first quad-word and continues towards the most significant end of the quad-word and then to higher memory addresses. The first bit of a quad-word is bit 7 of Byte 0 and the last bit is bit 0 of Byte 7.

12.10.3. BITBIT OPERATIONS

Using the GE’s BitBlt commands it is possible to implement the following six operations:

1) Rectangular Fill

2) Screen-to-screen BitBlt

3) Host-to-screen BitBlt

4) Packed Text

5) Microsoft Font Text

6) Line Segments

12.10.4. RECTANGULAR FILL

A rectangular fill operation is used to fill rectangular areas in the frame buffer with solid or patterned colours. The function performed during the fill operation is:

ROP ((Background), (Pattern), (Destination)) -> (Destination)

The specified rectangle is filled with the contents of the Background colour register, the pattern data and the existing destination contents, as modified by the ROP. The CPU provides the rectangle upper-left (Dst_XY) coordinates, the width and the height of the rectangle. Destination and pattern data can be anywhere in the frame buffer. ROP may be any of the 256 standard raster operations.

The Rectangular Fill operation is optimised to run at the memory bandwidth.

To perform a rectangular fill (except for the last write, order is unimportant):

Set the starting base address of the screen or region into Dst_base
Set the pitch of the screen or region into Dst_pitch
Set Pixel_depth.X_dir, Pixel_depth.Y_src_dir, and Pixel_depth.Y_dst_dir to ‘0’,
Set Width.Width to one less than the width of the rectangular area, in Bytes
Set Height.Height to one less than the height of the rectangular area, in Bytes
Set ROP.ROP to the raster operation, ROP.Expand to “0”, and ROP.Transparent OPAQUE
Set ROP.Source to CONSTANT_FILL
Set the Background colour register to the colour to be painted
Set the Pixel_depth for the number of Bytes-per-pixel
If a pattern is to be applied, set the Pattern register with the starting address of the pattern data
Write the upper-left coordinates of the area to be filled into the Dst_XY register. This write starts the BitBlt.

Frequently, some of these parameters will already be set up from prior operations, such as Dst_pitch, Background colour, and Pixel_depth. These will not need to be written each time.

The implementation of this operation is performance optimised and can drive the DRAM buffer at full bandwidth. Thus, resulting pixels are computed in groups of 32 bits, to allow one 64-bit result every two

video domain clock cycles. As the memory subsystem supports separate Byte-write enables, the first and last 64-bit write in each scan line can be performed without requiring a read-modify-write cycle.

12.10.5. SCREEN-TO-SCREEN BITBIT

The Screen-to-screen BitBlt operation is used to copy data from one rectangle in the frame buffer (either on-screen or off-screen areas) to another with the identical geometry. The pixel depth of the source region must match that of the destination region, or it may be a bitmap (if bitmap expansion is specified by setting ROP.Expand).

The function performed during the BitBlt operation is:

ROP ((Source), (Pattern), (Destination)) -> (Destination)

If these rectangular areas are overlapping, then the direction of the BitBlt must be carefully selected:

* *Source region is below the destination:*

> BitBlt should be done from the upper-left-hand corner and progress downwards,

> the X_dir, Y_src_dir and Y_dst_dir fields of the Pixel_depth register should be “0”.

* *Source region is above the destination:*

> BitBlt should be done from the lower-right-hand corner and progress upwards,

> the X_dir, Y_src_dir and Y_dst_dir fields of the Pixel_depth register should be “1”.

The Source, destination and pattern data can be anywhere in the frame buffer. The ROP may be any one of the 256 standard raster operations.

To perform a Screen-to-screen BitBlt (except for the last write, order is unimportant):

Set the starting base address of the destination screen or region into Dst_base

Set the pitch of the destination screen or region into Dst_pitch

Set the starting base address of the source region into Src_base

Set the pitch of the source region into Src_pitch

Set Pixel_depth.X_dir, Pixel_depth.Y_src_dir, and Pixel_depth.Y_dst_dir to ‘0’ or to ‘1’, depending upon direction of the BitBlt

Set Width.Width to one less than the width of the destination rectangular area, in Bytes

Set ROP.ROP, ROP.Expand and ROP.Transparent to the appropriate values

Set ROP.Source to SCREEN

If Colour Transparency mode is set, the set Background colour register to the colour to be made transparent

If a pattern is to be applied, set the Pattern register with the starting address of the pattern data

If ROP.Expand is set, then also set the Foreground and Background colour registers

Write the starting coordinates of the source area to be copied into the Src_XY register

Write the starting coordinates of the destination area into the Dst_XY register. This write starts the BitBlt.

Frequently, some of these parameters will already be set up from prior operations, such as Dst_pitch, Foreground colour, Background colour, and Pixel_depth. These will not need to be written each time.

The implementation of this operation is performance optimised and can drive the DRAM buffer at full bandwidth during constant fills. Thus, resulting pixels are computed in groups of 32 bits, to allow one 64-bit result every two graphics clock domain cycles. As the memory subsystem supports separate Byte-write enables, the first and last 64-bit write in each scan line can be performed without requiring a read-modify-write cycle.

12.10.6. HOST-TO-SCREEN BITBIT

The Host-to-screen BitBlt is used to copy data from the Host CPU to the frame buffer (either on-screen or off-screen areas). Note that if the CPU has built a rectangle in the frame buffer memory area with the Host data, then the Screen-to-Screen BitBlt operation can be used instead of this operation.

The pixel depth of the Host data must match that of the Destination region, unless it is a bitmap (if bitmap expansion is specified).

The function performed during the BitBlt is:

ROP((Host),(Pattern),(Destination))->(Destination).

The host area data is supplied by the CPU, which writes its data into the Data Port. The destination and pattern data can be anywhere in the frame buffer.

ROP may be any one of the 256 standard raster operations.

The CPU specifies the number of least significant Bytes of the first double-word that should be discarded, via the least significant three bits of the X field in the Src_XY register. The GE then merges Bytes of two double-words at a time, in order to build a double-word to operate on. For example, if the X field was set to 3, then the last Byte of the first double-word and the first three Bytes of the second double-word would be combined to form the first Host data double-word:

	Double Word 1				Double Word 0			
Byte:	7	6	5	4	3	2	1	0

Byte:	6	5	4	3

The CPU must provide the number of words required for Height * Width pixels. At the end of a scan line, the GE will discard the excess Host Bytes or bits that may be left in the last double-word and advance to the next scan line, unless Src_pitch is set to 0. In this case, data for adjacent scan lines are contiguous in the host data stream.

To perform a Host-to-screen BitBlt (except for the last write, order is unimportant):

Set the starting base address of the destination screen or region into Dst_base

Set the pitch of the destination screen or region into Dst_pitch

Set Src_pitch to 0 if packed source data or to a non-zero value, otherwise

Set Pixel_depth.X_dir, Pixel_depth.Y_src_dir, and Pixel_depth.Y_dst_dir to ‘0’

Set Width.Width to one less than the width of the destination rectangular area, in Bytes

Set ROP.ROP, ROP.Expand and ROP.Transparent to the appropriate values

Set ROP.Source to HOST

If Colour Transparency mode is set, set the Background colour register to the colour to be made transparent

Set the Pixel_depth for the number of Bytes-per-pixel

If a pattern is to be applied, set the Pattern register with the starting address of the pattern data

If ROP.Expand is set, then also set the Foreground and Background colour registers

Set Src_XY to indicate alignment of the Host data (only the least 3 significant bits are important)

Write the starting coordinates of the destination area into the Dst_XY register. This writes the BitBlt.

The image data is then written to the GE’s Data Port, if the GE is unable to accept additional data the CPU will be held off (invisibly to the software)

Frequently, some of these parameters will already be set up from prior operations, such as Dst_pitch, Foreground colour, Background colour, and Pixel_depth. These will not need to be written each time.

12.10.7. PACKED TEXT

The Packed Text operation is used to efficiently expand packed bitmap fonts into full colour representations in the frame buffer (either on-screen or off-screen areas). This operation is implemented as a Host-to-screen BitBlt with bitmap expansion and packed source data. The next section discusses how to handle Microsoft Font Text operations.

The function performed during the Packed Text operation is:

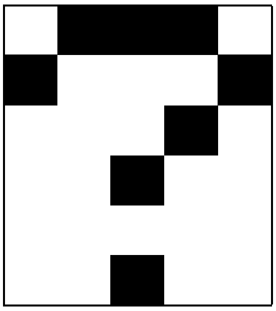
ROP ((Host), (Pattern), (Destination)) -> (Destination).

The Host packed bitmap data is supplied by the CPU via writes to the Data Port and is organised as double-words containing 32 bits of information. Each bit corresponds to a pixel. This data is expanded into Background and Foreground colours, unless the bitmap expansion transparent mode is on. If the transparent mode is set, then Host data bits of ‘0’ suppress any changes to the corresponding destination pixels. The Destination and Pattern can be anywhere in the frame buffer.

ROP may be any one of the 256 standard raster operations.

In a standard bitmap, the start of each scan line is aligned to a pitch-specified boundary. This is acceptable for wide bitmaps, however text font bitmaps are usually not very wide. To increase the amount of information provided to the GE per Host write, the Text operands are bit-packed. Each 32-bit write contains only useful font data, except possibly for the trailing bits of the last write.

For example, question mark character might appear in a fictitious font as:



Or in Binary form:

0	1	1	1	0
1	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	0	0	0
0	0	1	0	0

This would appear in memory as in Table 12-12.

In this example, the entire character bitmap fits into a single 32-bit double-word. This is a big saving when compared to having to send one 32-bit double-word for each font row. Note that two bits of don’t cares exist at the (top) end of the double word. Since this character is five bits wide and six lines high, it only needs 30 bits of storage. The remaining two bits will not be displayed.

After setting up the registers, the CPU writes the Host data, in 32-bit quantities, to the Data Port.

If a Pattern is applied to the text operation, a row of the pattern data will be read at the start of each character scan line.

Table 12-12. Bit Representation

top line					bottom line
01110	10001	00010	00100	00000	00100
Increasing memory addresses ------>					

Breaking this up into Bytes see Table 12-13.

Table 12-13. Byte Representation

	01110100	01000100	01000000	000100XX
Byte	0	1	2	3

Breaking this up into a double-word as in Table 12-14.

Table 12-14. Double Word Representation

	000100XX	01000000	01000100	01110100
Byte	3	2	1	0
Word	0		1	

To perform a Packed Text BitBlt (except for the last write, order is unimportant):

Set the starting base address of the destination screen or region into Dst_base

Set the pitch of the destination screen or region into Dst_pitch

Set the ROP.Packed to 1 for packed source data

Set Pixel_depth.X_dir, Pixel_depth.Y_src_dir, Pixel_depth.Y_dst_dir to ‘0’

Set Height to the height of the destination character, in scan lines

Set ROP.ROP, ROP.Expand and ROP.Transparent to the appropriate values

Set ROP.Source to HOST

Set the Pixel_depth for the number of Bytes-per-pixel

If a pattern is to be applied, set the Pattern register with the starting address of the pattern data

set the Foreground and Background colour registers

Set Src_XY to indicate alignment of the Host data (only the least 5 significant bits are important)

Write the starting coordinates of the destination area into the Dst_XY register. This writes the BitBlt.

The character font data is then written to the GE’s Data Port, if the GE is unable to accept additional data the CPU will be held off (invisibly to the software)

To draw the next character, its starting address (taking into account inter-character spacing) is written to the Dst_XY register, along with that character’s width encoded into the address of the Dst_XY register. This write can be done even if the GE is busy, as the Destination and Width registers are double buffered. The CPU then writes all the bitmap data that corresponds to the second character, the third Dst_XY/Width, the third bitmap data, etc.

Frequently, some of these parameters will already be set up from prior operations, such as Dst_pitch, Foreground colour, Background colour, and Pixel_depth. These will not need to be written each time.

12.10.8. MICROSOFT FONT TEXT

Microsoft fonts (consisting of 8-bit strips of a character) can be handled as simple 8-pixel-wide Host-to-screen BitBlts with bitmap expansion, but no packed data. The last strip of a character is handled in a different manner. The background colour for the last strip is first filled into its rectangular area. Then the strip data is drawn in transparent mode with the unused bits filled with zeros.

12.10.9. LINE SEGMENTS

The line segment operations are used to draw horizontal or vertical line segments. The segments are runs of pixels that start from a specified coordinate address (via the Dst_XY register) and whose length is specified in the address used when writing to the Dst_XY register.

The function performed during the line draw is:

ROP ((Background), (Pattern), (Destination)) -> (Destination)

ROP may be any one of the 256 standard raster operations.

Simple and complex curves can be efficiently drawn. The software on the CPU must generate all points or scan lines to be drawn and then use the GE to draw the line segments.

Two different types of line segments are supported: horizontal and vertical. For horizontal line segments, the Height register should be programmed to ‘0’, to indicate a single pixel high line. The length of the line segment (in Bytes) will then be stored into the Width register when the Dst_XY register is written to (the length is encoded into the Count field of the Dst_XY register’s address). For vertical lines, the Width register should be programmed to one less than the number of Bytes per pixel, to indicate a single pixel wide line. The length of the line segment is stored into the Height register when the Dst_XY register is written to.

It is possible to draw thicker line segments, by programming the Height register (for horizontal segments) or the Width register (for vertical segments) to other values.

For horizontal line segments, the X_dir, Y_src_dir and Y_dst_dir fields of the Pixel_depth register must be set to “0”.

The Background colour register should be set to the colour of the line segment to be drawn.

12.11. CURSOR SUPPORT

The GE supports a 64x64x2 cursor. The cursor is actually two 64x64x1 arrays: an AND array and an XOR array. For any given pixel that is within the cursor's active region, the displayed pixel depends on the frame buffer pixel, the AND array value, the XOR array value and the Cursor_colour0 and Cursor_colour1 registers as shown in Table 12-15.

The AND array is stored in off-screen memory, starting at Cursor. The XOR array is stored in off-screen memory starting at (Cursor + 512). Two 64-bit on-chip registers hold one scan line of each of these arrays. Before a scan line that possibly includes a cursor is displayed, these two registers are loaded from the appropriate off-screen locations.

Note that for 8-bit and 16-bit pixel depths, the above cursor operation is performed AFTER the data has been expanded by the colour look-up-table (LUT). Thus, the Inverted Frame Buffer Pixel is the complement of the full-colour pixel that would otherwise be displayed.

The cursor address (Cursor_XY) refers to the upper-left-hand corner of the cursor and specifies the distance, in pixels, from the upper-left-hand corner of the screen. So, if the cursor address were to be set to (0,0), then the entire cursor could be displayed in the upper-left-hand corner of the screen. The cursor active region thus may extend from:

(Cursor_XY.X, Cursor_XY.Y)

to

(Cursor_XY.X + 63, Cursor_XY.Y + height - 1)

as controlled by the Cursor Height register (CR29).

Note: To suppress cursor display, enter one more than the number of display scan lines into the Y field.

Table 12-15. Cursor Arrays

AND Value	XOR Value	Displayed Pixel
0	0	Cursor_colour0
0	1	Cursor_colour1
1	0	Frame Buffer Pixel
1	1	Inverted Frame Buffer Pixel

12.11.1. CURSOR HEIGHT REGISTER (RW)

CR29 Access = 0x3X4h/0x3X5h Regoffset = 029h

7	6	5	4	3	2	1	0
C XOR	CH						
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	C XOR	**Cursor XOR Pre/Post Look Up Table.** When this bit is set to one, the graphics cursor XOR operation is performed before the look up table. The default behaviour, when this bit is set to zero, is for the XOR operation to happen after the look up table. This is correct for 15, 16, 24 bit per pixel modes but not 8bpp.
Bits 6-0	CH	**Cursor height.** This field represents the vertical extent of the graphics cursor in scan lines. Setting this to zero effectively turns the graphics cursor off. Values greater than 40h (decimal 64) are meaningless and produce unpredictable results.

Programming notes:

There is no cursor width register - the width is always 64 pixels. If a narrower cursor is required, pad the bitmap on the right with transparent cursor colour (pad the AND plane with ‘1’s on the right and the XOR plane with ‘0’s).

12.11.2. CURSOR COLOUR 0 REGISTER A (RW)

CR2A Access = 0x3X4h/0x3X5h Regoffset = 02Ah

7	6	5	4	3	2	1	0
CC 0 R							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	CC 0 R	**Cursor Colour 0 Red.** These bits are the red component of cursor colour 0.

12.11.3. CURSOR COLOUR 0 REGISTER B (RW)

CR2B Access = 0x3X4h/0x3X5h Regoffset = 02Bh

7	6	5	4	3	2	1	0
CC 0 G							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	CC 0 G	**Cursor Colour 0 Green.** These bits are the green component of cursor colour 0.

12.11.4. CURSOR COLOUR 0 REGISTER C (RW)

CR2C Access = 0x3X4h/0x3X5h Regoffset = 02Ch

7	6	5	4	3	2	1	0
CC 0 B							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	CC 0 B	**Cursor Colour 0 Blue.** These bits are the blue component of cursor colour 0.

12.11.5. CURSOR COLOUR 1 REGISTER A (RW)

CR2D Access = 0x3X4h/0x3X5h Regoffset = 02Dh

7	6	5	4	3	2	1	0
CC 1 R							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	CC 1 R	**Cursor Colour 1 Red.** These bits are the red component of cursor colour 1.

12.11.6. CURSOR COLOUR 1 REGISTER B (RW)

CR2E Access = 0x3X4h/0x3X5h Regoffset = 02Eh

7	6	5	4	3	2	1	0
CC 1 G							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	CC 1 G	**Cursor Colour 1 Green.** These bits are the green component of cursor colour 1.

12.11.7. CURSOR COLOUR 1 REGISTER C (RW)

CR2F Access = 0x3X4h/0x3X5h Regoffset = 02Fh

7	6	5	4	3	2	1	0
CC 1 B							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	CC 1 B	**Cursor Colour 1 Blue.** These bits are the blue component of cursor colour 1.

12.11.8. GRAPHICS CURSOR ADDRESS REGISTER 0 (RW)

CR30 Access = 0x3X4h/0x3X5h Regoffset = 030h

7	6	5	4	3	2	1	0
CAA							Rsv
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-1	CAA	**Cursor AND Address Bits 15-9.** These bits represent bits 15-9 of the DRAM linear address of the cursor's AND mask. The cursor's XOR mask begins at this address + 512. This memory must be aligned on a 1 KByte boundary. For a discussion of DRAM linear addresses, see Section 12.6.
Bit 0	Rsv	**Reserved.** This bit should be written as zero.

Programming notes:

The cursor bitmap is ordered such that the top left hand corner of the cursor is represented by bit 7 of the Byte addressed by this field (AND) and bit 7 of the Byte at 512 plus this address (XOR plane). The next pixel right is represented by bit 6 of these Bytes and so on until the bottom right hand pixel is represented by bit 0 of the Byte located at this address plus 511 (AND) and bit 0 of the Byte at 1023 plus this address.

12.11.9. GRAPHICS CURSOR ADDRESS REGISTER 1 (RW)

CR31 Access = 0x3X4h/0x3X5h Regoffset = 031h

7	6	5	4	3	2	1	0
CAA							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0	CAA	**Cursor AND Address Bits 23-16.** These bits represent bits 23-16 of the DRAM linear address of the cursor's AND mask. For a discussion of DRAM linear addresses, see Section 12.6.

12.11.10. GRAPHICS CURSOR ADDRESS REGISTER 2 (RW)

CR32 Access = 0x3X4h/0x3X5h Regoffset = 032h

7	6	5	4	3	2	1	0
Rsv					CAA		
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-3	Rsv	**Reserved.** These bits should be written as zero.
Bits 2-0	CAA	**Cursor AND Address Bits 26-24.** These bits represent bits 26-24 of the DRAM linear address of the cursor's AND mask. For a discussion of DRAM linear addresses, see Section 12.6.

13. LINE DRAW ENGINE

13.1. FEATURES

- Supports anti-aliased line rendering.
- Clipping of lines supported.
- Special tile cache supported for high performance.
- Special software reset support.
- All registers are double buffered unless otherwise stated.
- Support for 16-bit (555 & 565) colour format.
- Support for 24-bit (888) true colour format.
- Complete pipeline structure for rendering pixel after pixel for different lines operating at 100 MHz.

13.1.1. DOUBLE BUFFERING

The Line Draw Engine (LDE) can be programmed to render two lines, one after the other i.e. the software can pipeline two commands for the LDE. Once the parameters for the first line are programmed and the command is issued, all the LDE double buffered registers are available for programming and issuing the command for the next line. For further information about pipeline operation Section 13.5.22. WNA and WDI bits.

13.2. MEMORY ADDRESS SPACE

In STPC architecture a 16 MByte memory space is allocated for extended graphics and video functions. The start of the 16 MByte memory space is programmable using the GBASE (in CRTC regs).

This extended graphics memory is divided into four 4 MByte regions as shown in Figure 13-1. The second 4 MBytes (Region 1) is used for memory mapped registers needed for extended graphics and video functionality. This 4 MByte region is further subdivided into sixteen 256 KByte regions. The fourth 256 KBytes (Region 3) is further subdivided into four 64 KBytes and each of these 64 KBytes is subdivided into sixteen 4 KBytes. The Line Draw Engine (LDE) uses the 6th 4 KByte region for its register access.

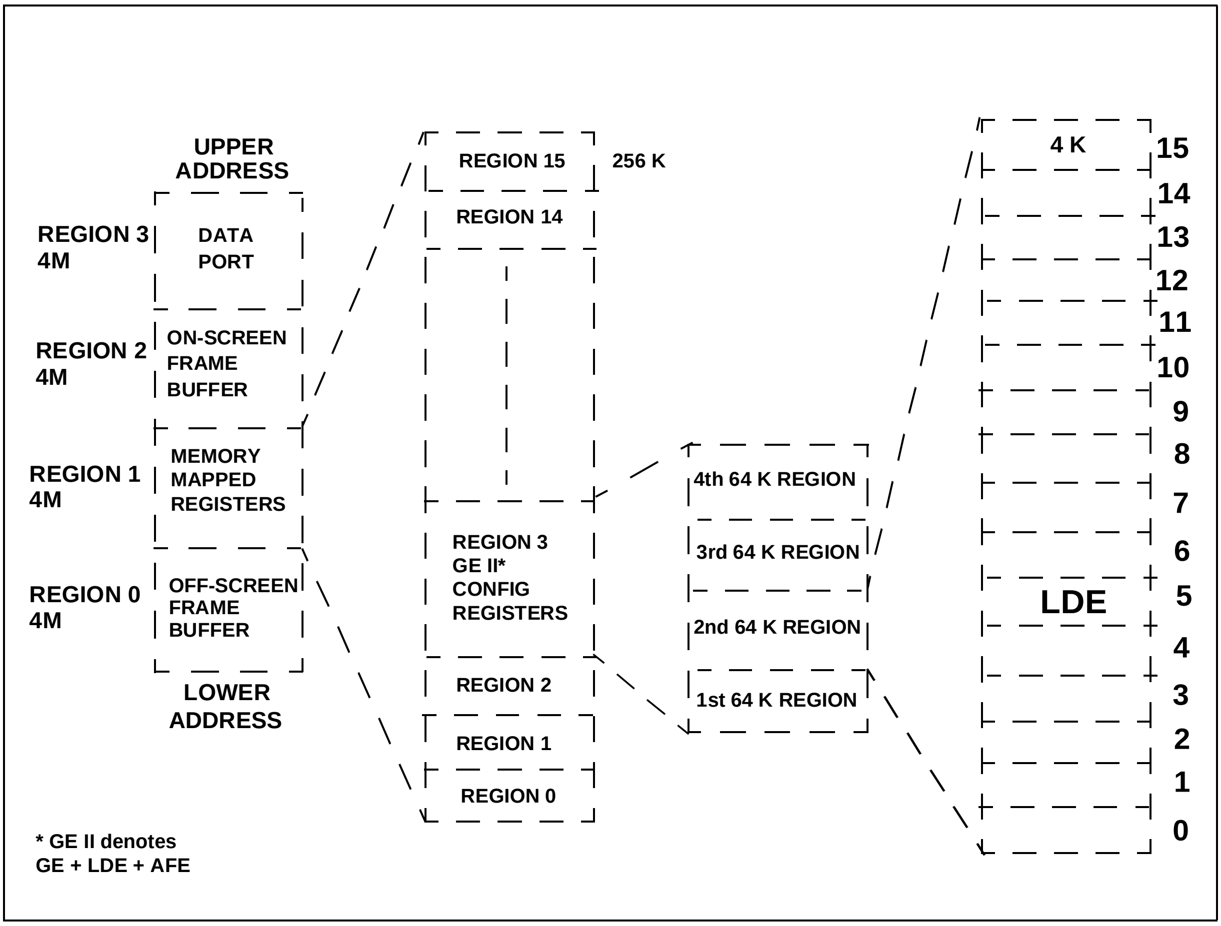


Figure 13-1. Extended Graphics Memory Map

13.3. REGISTER ACCESS

Refer to Figure 13-1.

Table 13-1. Access Address Space

00001	<GBASE>	01	0011	01	0101	<REG>
31 27	26 24	23 22	21 18	17 16	15 12	11.... 0
		4 MByte Region 1 (memory mapped region)	256 KByte Region 3	2nd 64 KByte region	6th 4 KByte region	
Default value after reset = GBASE + 4D5000h						

<REG> specifies the offset of the register to be accessed.

Configuration registers are double buffered with the following exceptions:
LUT_Reg, Misc_Reg, FBBaseAddr, FBPitchMulti_Reg, FBPitch_Reg, ClipMaxCordi_Reg<
Status_Reg, Soft_Reset_Reg, Performance_Reg, Flush_Reg and ClipMinCordi_Reg

The CPU can program all the registers and can issue a command. Once the status bit [31] has a value ‘1’, indicating “LDE_Busy”, all the registers can be programmed for the next command. The CPU can issue another command in the pipeline. At this point, since the first command is not yet over, the status bit [30] has a value ‘1’, indicating “no more writes to registers are allowed” (since both the buffers are full). Then if the CPU tries to update any register, the cycle is terminated properly but the write data is ignored and the status bit [29] has a value ‘1’, indicating “overwrite data ignored”. Reads to any register can be made at any time. Reads to the configuration register access data from the un-buffered set of registers.

13.4. OPERAND FRAME BUFFER ADDRESSES

Internally, LDE implements the following equation to generate the required Frame Buffer linear address from X, Y, FBBaseAddr and FBPitchMulti registers.

Frame Buffer linear address = Frame Buffer base + X * (2 or 3) + (Y * pitch)

Pitch is the number of bytes used in the frame buffer to display one scan line.

The multiplication of the pitch is done by hardware shifts and add. The pitch value is actually specified as group of 4 shift codes. For each non-zero shift code, the Y address is shifted by the corresponding number of bits and then added. X is multiplied by 2 if PixelDepth = 1 (2 bytes/pixel), or 3 if PixelDepth = 2 (3 bytes/ pixel). The shift values supported are:

Table 13-2. Shift Values Supported

Value	FBPitchMulti[10:9] shift 3	FBPitchMulti[8:6] shift 2	FBPitchMulti[5:3] shift 1	FBPitchMulti[2:0] shift 0
000	0	0	0	0
001	1024 * dstY	32 * dstY	0	0
010	2048 * dstY	64 * dstY	64 * dstY	0
011	4096 * dstY	128 * dstY	128 * dstY	128 * dstY
100	n/a	256 * dstY	256 * dstY	256 * dstY
101	n/a	512 * dstY	512 * dstY	512 * dstY
110	n/a	0	1024 * dstY	1024 * dstY
111	n/a	0	0	2048 * dstY

The Frame Buffer base addresses must be aligned to 32 bytes (i.e. the five least significant bits of the address must be zeros). The supported pitches (in bytes) are:

Table 13-3. Pitches Supported

0	544	1216	1792	2432	3136	4160	4704	5408	6432
32	576	1248	1824	2464	3168	4192	4736	5440	6464
64	608	1280	1856	2496	3200	4224	4768	5504	6528
96	640	1312	1920	2560	3232	4256	4800	5632	6656
128	672	1344	2048	2592	3264	4288	4864	5664	6688
160	704	1376	2080	2624	3328	4320	4896	5696	6720
192	768	1408	2112	2656	3360	4352	4928	5760	6784
224	800	1440	2144	2688	3392	4384	4992	5888	6912
256	832	1472	2176	2720	3456	4416	5120	6144	7168
288	896	1536	2208	2752	3584	4448	5152	6176	7200
320	1024	1568	2240	2816	3616	4480	5184	6208	7232
352	1056	1600	2272	2848	3648	4512	5216	6240	7296
384	1088	1632	2304	2880	3712	4544	5248	6272	7424
416	1120	1664	2336	2944	3840	4608	5280	6304	7680
448	1152	1696	2368	3072	4096	4640	5312	6336	
512	1184	1728	2400	3104	4128	4672	5376	6400	

13.5. REGISTER DESCRIPTION

13.5.1. XSTART REGISTER

This register specifies the start X location for the line. The CRTC display can be thought of as a 2D grid of pixels, where this Xstart value points to the pixel in the X direction. The Xstart value can be negative or positive. This register should always be programmed with a valid value since this register, along with the Xend, Ystart and Yend registers, determines the visible dimension of the line.

XSTART_REG Access: = GBASE + 4D5000h Regoffset = 0x00h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
XS															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XS															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**XS**	**Xstart address**

This is a read/write register.

13.5.2. YSTART REGISTER

This register specifies the start Y location for the line. The CRTC display can be thought of as a 2D grid where this Ystart value points to the byte location in the Y direction. The Ystart value can be negative or positive. This register should always be programmed with a valid value since this register, along with the Xstart, Xend and Yend registers, determines the visible dimension of the line.

YSTART_REG Access: = GBASE + 4D5000h Regoffset = 0x04h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
YS															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
YS															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**YS**	**Ystart address**

This is a read/write register.

13.5.3. XEND REGISTER

This register specifies the end X location for the line. The CRTC display can be thought of as a 2D grid where this Xend value points to the byte location in the X direction. The Xend value can be negative or positive. This register should always be programmed with a valid value since this register, along with the Xstart, Ystart and Yend registers, determines the visible dimension of the line.

XEND_REG Access: = GBASE + 4D5000h Regoffset = 0x08h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
XE															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XE															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**XE**	**Xend address**

This is a read/write register.

13.5.4. YEND REGISTER

This register specifies the end Y location for the line. The CRTC display can be thought of as a 2D grid where this Yend value points to the byte location in the Y direction. The Yend value can be negative or positive. This register should always be programmed with a valid value since this register, along with the Xstart, Xend and Ystart registers, determines the visible dimension of the line.

YEND_REG Access: = GBASE + 4D5000h Regoffset = 0x0Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
YE															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
YE															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**YE**	**Yend address**

This is a read/write register.

13.5.5. DX REGISTER

This register specifies the difference between the Xend and Xstart of a line. The register needs to be programmed only if subpixel is enabled.
When subpixel is OFF, dx is calculated from Xend-Xstart.
When subpixel is ON, Xstart and Xend must not be used to calculate DX and DY since they are still in pixel coordinates whereas the DX value is in subpixel pixel coordinates.
The dx value can be negative or positive.

DX_REG Access: = GBASE + 4D5000h Regoffset = 0x10h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DX															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DX															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**DX**	**DX value**

This is a read/write register.

13.5.6. DY REGISTER

This register specifies the difference between the Yend and Ystart of a line. The register needs to be programmed only if subpixel is enabled.
When subpixel is OFF, dy is calculated from Yend-Ystart.
When subpixel is ON, Ystart and Yend must not be used to calculate DX and DY since they are still in pixel coordinates whereas the DY value is in subpixel pixel coordinates.
The dy value can be negative or positive.

DY_REG Access: = GBASE + 4D5000h Regoffset = 0x14h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DY															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DY															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**DY**	**DY value**

This is a read/write register.

13.5.7. SCALED DX REGISTER

Always program this register.
The formula for the scaling is:Scaledx_Reg = (dx << 15 - dx << 10) / abs(3*(xmajor ? dx : dy))

The primary idea of scaling is to adjust for floating point calculations. Hardware values are integer based so the software rounds or scales up the mantissa part of the fraction.

SCALEDDX_REG Access: = GBASE + 4D5000h Regoffset = 0x18h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SDX															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDX															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**SDX**	**Scaled dx value**

This is a read/write register.

13.5.8. SCALED DY REGISTER

Always program this register.
The formula for the scaling is:Scaleddy_Reg = (dy << 15 - dy << 10) / abs(3*(xmajor ? dx : dy))

The primary idea of scaling is to adjust for floating point calculations. Hardware values are integer based so the software rounds or scales up the mantissa part of the fraction.

SCALEDDY_REG Access: = GBASE + 4D5000h Regoffset = 0x1Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SDY															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDY															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**SDY**	**Scaled dy value**

This is a read/write register.

13.5.9. FXY REGISTER

This register specifies the Fxy value for a line and is used in calculating the initial decision variable of a line. The register needs to be programmed only if subpixel is enabled. The Fxy value can be negative or positive.

FXY_REG Access: = GBASE + 4D5000h Regoffset = 0x20h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FXY															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FXY															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**FXY**	**Fxy value**

This is a read/write register.

13.5.10. SCALED FXY REGISTER

This register specifies scaled Fxy value for a line, the value of which is given in Section 13.6. The register needs to be programmed only if anti-alias & subpixel are enabled. The Scaledfxy value can be negative or positive.

SCALEDFXY_REG Access: = GBASE + 4D5000h Regoffset = 0x24h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SFXY															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SFXY															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**SFXY**	**Scaled Fxy value**

This is a read/write register.

13.5.11. END POINT CORRECT REGISTER

This register holds the end point correction for a line.

If EPC is not desired then all the ‘Correct’ values should be programmed with the value 0x10h

If EPC is desired then the following cases determine if the EPC register should be programmed or not.

CASE 1: If line width = 1 then whether anti-alias is ON or OFF this register should be programmed with correct EPC values.

CASE 2: If line width > 1 then if anti-alias is ON program the correct EPC value in this register.

CASE 3: If line width >1 then if anti-alias is OFF programming this register is NOT important.

CORRECT_REG Access: = GBASE + 4D5000h Regoffset = 0x28h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv			EPC2					Rsv			EPC1				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv			SPC2					Rsv			SPC1				
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-29	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 28-24	**EPC2**	**End Pixel 2 Correct**
Bits 23-21	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 20-16	**EPC1**	**End Pixel 1 Correct**
Bits 15-13	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 12-8	**SPC2**	**Start Pixel 2 Correct**
Bits 7-5	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 4-0	**SPC1**	**Start Pixel 1 Correct**

This is a read/write register.

Programming note:

The sequence of pixel correct values used varies with the line length. It is important to program the right correct value depending on the line length as follows:

Line Length	Pixel Drawn on Screen	Correct Value Used
>= 4	1st 2nd 2nd last last	SPC1 SPC2 EPC2 EPC1
= 3	1st 2nd last	SPC1 SPC2 EPC1

Line Length	Pixel Drawn on Screen	Correct Value Used
= 2	1st 2nd	SPC1 SPC2
= 1	1st	SPC1

Following are some examples of how the line appears with EPC in various configurations

For all the examples below the line draw parameters are:
foreground colour is RED, background colour is WHITE
SPC1 = 0x06h, SPC2 = 0x0Ah
EPC1 = 0x0Ah, EPC2 = 0x06h

1) Anti-alias is ON, EPC is desired and line width = 1:

2) Anti-alias is OFF, EPC is desired and line width = 1:

3) Anti-alias is ON, EPC is desired and line width > 1:

4) Anti-alias is OFF, EPC is desired and line width >1:

13.5.12. LUT REGISTER

This register is used to program the look-up table (LUT) with the alpha blend values. The LUT is a combination of 32 8-bit registers. To initialize the complete LUT 32 8-bit writes need to be performed. With every LUT write the pointer increments and points to the next LUT 8-bit register. At any point the LUT write pointer can be reset by writing a ‘1’ to bit [31] in the Misc register (section 13.5.16.). After 32 8-bit writes or after reset, the LUT write pointer automatically points to the first LUT location.

For LUT reads there is a read pointer which points to the first of four LUT registers. A single LUT read returns data from four LUT registers and increments the pointer by four ready for the next read (i.e. the first LUT read will return data from LUT registers 0, 1, 2 and 3, the next LUT read will return data from LUT registers 4, 5, 6 and 7, and so on). At any point the LUT read pointer can be reset by writing a ‘1’ to bit [29] in the Misc register. After 32 8-bit reads or after reset, the LUT write pointer automatically points to the first LUT location. LUT reads are double word aligned.

LUT register write/read can be done in any order.

LUT_REG Access: = GBASE + 4D5000h Regoffset = 0x2Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv								LUT							
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-8	**Rsv**	**Reserved**
Bits 7-0	**LUT**	**LookUp Table**

This is a read/write register.

13.5.13. LINE WIDTH REGISTER

This register specifies the width of a line. The line width register value can only be positive.

The line width value indicates the number of pixels which will be highlighted for the line draw operation.
If anti-alias is OFF then:
line width of 1 implies a 1 pixel wide line,
line width of 2 implies a 2 pixel wide line.
If anti-alias is ON then:
line width of 1 implies a 3 pixel wide line,
line width of 2 implies a 4 pixel wide line and so on.

LINEWIDTH_REG GBASE + 4D5000h Regoffset = 0x30h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv					LW										
Default value after reset = 0x00000001h															

Bit Number	Mnemonic	Description
Bits 31-11	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 10-0	**LW**	**Line Width**

This is a read/write register.

13.5.14. LINE COLOUR REGISTER

This register defines the foreground line colour. The programming value of the colour is the same for both 16-bit and 24-bit per pixel modes. The only difference is that internally the LDE uses all 8 bits of RGB in 24-bit mode and the MSB 565/555 bits of RGB in 16-bit 565/555 mode.

LINECOLOUR_REG GBASE + 4D5000h Regoffset = 0x34h

<table>
<tr><th colspan="16">24-bit (888) pixel format</th></tr>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="8">Rsv</td><td colspan="8">R</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="8">G</td><td colspan="8">B</td></tr>
<tr><th colspan="16">16-bit (565) pixel format</th></tr>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="8">Rsv</td><td colspan="5">R</td><td colspan="3">Rsv</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="6">G</td><td colspan="2">Rsv</td><td colspan="5">B</td><td colspan="3">Rsv</td></tr>
<tr><th colspan="16">16-bit (555) pixel format</th></tr>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="8">Rsv</td><td colspan="5">R</td><td colspan="3">Rsv</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="5">G</td><td colspan="3">Rsv</td><td colspan="5">B</td><td colspan="3">Rsv</td></tr>
<tr><td colspan="16">Default value after reset = 0x00000000h</td></tr>
</table>

Bit Number	Mnemonic	Description
Bits 31-24	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 23-16 24-bit **Bits 23-19 16-bit**	**R**	**Red**
Bits 18-16 16-bit	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 15-8 24-bit **Bits 15-10 16-bit (565)** **Bits 15-11 16-bit (555)**	**G**	**Green**
Bits 9-8 16-bit (565) **Bits 10-8 16-bit (555)**	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 7-0 24-bit **Bits 7-3 16-bit**	**B**	**Blue**
Bits 2-0 16-bit	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.

This is a read/write register.

13.5.15. BACK COLOUR REGISTER

Fort a full explanation refer to section 13.5.14. "Line ColoUr Register".

.

BACKCOLOUR_REG Access: = GBASE + 4D5000h Regoffset = 0x38h

<table>
<tr><th colspan="16">24-bit (888) pixel format</th></tr>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="8">Rsv</td><td colspan="8">R</td></tr>
<tr><td colspan="16"></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="8">G</td><td colspan="8">B</td></tr>
<tr><th colspan="16">16-bit (565) pixel format</th></tr>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="8">Rsv</td><td colspan="5">R</td><td colspan="3">Rsv</td></tr>
<tr><td colspan="16"></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="6">G</td><td colspan="2">Rsv</td><td colspan="5">B</td><td colspan="3">Rsv</td></tr>
<tr><th colspan="16">16-bit (555) pixel format</th></tr>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="8">Rsv</td><td colspan="5">R</td><td colspan="3">Rsv</td></tr>
<tr><td colspan="16"></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="5">G</td><td colspan="3">Rsv</td><td colspan="5">B</td><td colspan="3">Rsv</td></tr>
<tr><td colspan="16">Default value after reset = 0x00000000h</td></tr>
</table>

Bit Number	Mnemonic	Description
Bits 31-24	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 23-16 24-bit **Bits 23-19 16-bit**	**R**	**Red**
Bits 18-16 16-bit	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 15-8 24-bit **Bits 15-10 16-bit (565)** **Bits 15-11 16-bit (555)**	**G**	**Green**
Bits 9-8 16-bit (565) **Bits 10-8 16-bit (555)**	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 7-0 24-bit **Bits 7-3 16-bit**	**B**	**Blue**
Bits 2-0 16-bit	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.

This is a read/write register.

13.5.16. MISC REGISTER

This is a multi purpose register controlling the features described below.

Note:

Writing ‘1’ to any of the three MSB bits [31-29] has no affect on bits [28-0]. The contents of bits [28-0] are therefore irrelevant when resetting the LUT read/write pointers or force flushing the tile RAMs.

MISC_REG Access: = GBASE + 4D5000h Regoffset = 0x3Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WCR	FF	RCR	Rsv											PC	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv	BM							Rsv	EOG	PME	MD	PD		AA	SP
Default value after reset = 0x00030000h															

Bit Number	Mnemonic	Description
Bit 31	**WCR**	**LUT Write Counter Reset** 1 = reset the LUT write counter. This is a self resetting bit and will read as ‘0’. The LUT is a set of 32 8-bit registers which are programmed through one address (refer to section 13.5.12. "LUT Register"). This register is NOT double buffered and takes effect as soon as it is written. Writing ‘1’ to this bit does not affect the programmed values of bits [28-0].
Bit 30	**FF**	**Tile RAM Force Flush** 1 = flush the tile RAMs immediately. This is a self resetting bit and will read as ‘0’. Writing ‘1’ to this bit does not affect the programmed values of bits [28-0]. This is one of the ways to force flush the tile RAMs. For the alternative way refer to section 13.5.25. "Flush Count Register". Note: Extreme care should be taken before using this feature as mis-calculated use will render wrong data. Always make sure that LDE is not busy and give a time gap before setting this bit to ‘1’. See ‘programming note’ on page 386
Bit 29	**RCR**	**LUT Read Counter Reset** 1 = reset the LUT read counter so that the next LUT read will be provided from the first LUT location (refer to section 13.5.12. "LUT Register"). This is a self resetting bit and will read as ‘0’. This register is NOT double buffered and takes effect as soon as it is written. Writing ‘1’ to this bit does not affect the programmed values of bits [28-0].
Bits 28-18	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.

Bits 17-16	**PC**	**Priority Control** These bits control the priority for the LDE SDRAM i/f in the GE II SDRAM bus i/f arbiter. The priorities are: lowest priority 00 < 01 < 10 < 11 highest priority. The GE II SDRAMC i/f arbiter implements a special bus parking logic for LDE: If LDE gets the SDRAMC i/f and the tile RAMs still have valid data in them (tile RAMs are dirty) then the arbiter will park the SDRAMC i/f bus to LDE until the LDE's tile RAMs are not dirty, i.e. the tile RAMs are flushed to the Frame Buffer. This will prevent other agents from accessing the same Frame Buffer locations which have already been cached in the tile RAMs of the LDE.
Bit 15	**Rsv**	**Reserved** - Read will return '0's for these bits.
Bits 14-8	**BM**	**Blend Mode** This defines the Blend mode for merging foreground and background colours. Blending is done in three stages as shown by Tables 13-4 to 13-6. Note: Blend mode should not be reprogrammed to a different value if the tile RAMs are dirty.
Bit 7	**Rsv**	**Reserved** - Read will return '0's for these bits.
Bit 6	**EOG**	**EOG for every address change** 1 = enable the SDRAMC i/f to output EOG with every change of address, i.e. at every tile RAM's row access (read/write). Hence for one tile RAM fill there will be 8 EOGs. 0 = EOG will be asserted at the end of tile fill. Hence for one tile RAM fill there will be 1 EOG.
Bit 5	**PME**	**Performance Meter Enable** 1 = performance metering is enabled. An internal counter counts the number of host clocks taken to perform one line draw operation. The performance register can be read at the end of the line draw operation to check the value. This bit is double buffered and therefore the performance metering can be switched on/off for different lines in the pipeline. While performance metering remains enabled: - Every time a new line draw command is initiated with the performance metering enabled, the old performance metering value will be destroyed. - This register can be read on the fly, but the correct value will only be returned after the end of the current line draw operation and before the start of the next line draw operation.
Bit 4	**MD**	**Mode 555** This bit is effective only when 16-bit per pixel format is selected, i.e. pixel depth = '01'. 1 = mode 555 (RGB) 0 = mode 565 (RGB) colour format. Note: This bit is not double buffered and should not be reprogrammed to a different value if the LDE is BUSY or if the tile RAMs are dirty. Also this bit should be in sync with the CRTC mode of operation.

Bits 3-2	**PD**	**Pixel Depth** Specifies the pixel depth (number of bits per pixel). Supported values are: 00 - not defined. 01 - 2 bits per pixel, RGB 565/555 mode only (16-bit format). 10 - 3 bits per pixel, RGB 888 mode only (32-bit format). 11 - not defined. Notes: 1) 8 and 32-bit pixel formats are not supported. 2) These bits are not double buffered and should not be reprogrammed to a different value if the LDE is BUSY or if the tile RAMs are dirty. Also these bits should be in sync with the CRTC mode of operation.
Bit 1	**AA**	**Anti-Alias** 1 = enable anti-alias for the line. The anti-alias attribute can be modified for each line. This bit is double buffered. When anti-alias is turned ON the line will be two pixels wider than the value programmed in the line width register. The extra two pixels will be shaded or blended with the background colour register or the screen background colour depending on the blendmode register programming.
Bit 0	**SP**	**Sub Pixel** 1 = enable the subpixel for the line. The subpixel attribute can be modified for each line. This bit is double buffered. When subpixel is enabled the dx and dy register values should be multiplied by 8 and programmed to the respective registers.

This is a read/write register.

Table 13-4. Blend Stage 1

Bit 9 Blend DEST	Bit 8 Blend BACK	Action
0	0	Blend colour is BLACK
0	1	Blend colour is background colour register
1	0	Blend colour is screen colour
1	1	Blend colour is background colour register

Table 13-5. Blend Stage 2

Bit 12 Blend SUB	Bit 11 Blend REV	Bit 10 Blend ALPHA	Action
0	0	0	No blend mode selected: Blend colour not used
0	0	1	Modify only the line colour: Blend colour not used
0	1	0	Blend with Reverse Alpha
0	1	1	Blend with Reverse Alpha

Table 13-5. Blend Stage 2

Bit 12 Blend SUB	Bit 11 Blend REV	Bit 10 Blend ALPHA	Action
1	0	0	Blend Sub
1	0	1	Blend Sub
1	1	0	Blend Rev
1	1	1	Blend Rev

Table 13-6. Blend Stage 3

Bit 15 Reserved	Bit 14 Blend MAX	Bit 13 Blend ADD_DEST	Action
	0	0	No blending
	0	1	Blend Add Dest
	1	0	Blend MAX
	1	1	Blend Add Dest

Programming note:

The LDE uses two tile RAMs which act as cache RAMs for the Frame Buffer data. An arbiter for GE, AFC and LDE arbitrates which of these have contact with the memory controller for Frame Buffer accesses. By default LDE has highest priority (modified by bits [17-16] of Misc_Reg). Also, since the tile RAMs prefetch Frame Buffer data, if a tile is dirty then the arbiter is locked to the LDE, preventing the AFC and GE contacting the memory controller bus. Only after a tile RAM flush (automatic or forced) will the tile dirty status be cleared so that the arbiter can resume arbitrating between all three agents.

Extreme care must be taken when using the force flush as is shown by the following scenario:

GE and LDE are being used by the OS.
LDE is drawing a line and the automatic flush register is not programmed.
The software forces a tile flush, without checking the LDE busy bit in the Status register, and now thinks that the tiles are no longer dirty.
The LDE, however, had not finished drawing the line so as the drawing process continues a tile RAM becomes dirty and the arbiter becomes locked to the LDE.
If the GE has started an operation it will be unable to complete it (being locked out from the memory controller bus) but the software will be waiting for completion.
Thus the system will hang, waiting for the GE to complete an operation.

13.5.17. FBBASE ADDRESS REGISTER

This register specifies the starting DRAM linear physical address of the destination operand (aligned to a 32 byte boundary).

Note:

This register is not double buffered and its value cannot be changed if the LDE is busy or the tile RAMs are dirty.

FBBASEADDR Access: = GBASE + 4D5000h Regoffset = 0x40h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FBBA															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FBBA											zeros				
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-5	**FBBA**	**FBbaseAddr** Base SDRAM linear address of the destination operand.
Bits 4-0	**zeros**	**Zeros** (32 byte aligned)

This is a read/write register.

13.5.18. FRAME BUFFER SHIFTS REGISTER

This register specifies the multiplication factors to calculate the number of bytes needed to advance from a pixel in one scan line of the Frame Buffer to the corresponding pixel in the next scan line. The value is always positive.

Note:

This register is not double buffered and its value cannot be changed if the LDE is busy or the tile RAMs are dirty.

Calculation:
Pitch value is specified as the number of bytes in one scan line.
For example, if there are 640 pixels in one scan line then:
 if each pixel is represented by 2 bytes/pixel, pitch = (640 x 2) = 1280 bytes
 if each pixel is represented by 3 bytes/pixel, pitch = (640 x 3) = 1920 bytes.
Use Table 13-2. section 13.4. to calculate the value of shift 0, shift 1, shift 2 and shift 3 in such a way that the sum of the shifts equals the above calculated bytes in one scan line.

FBPITCHMULTI_REG Access: = GBASE + 4D5000h Regoffset = 0x44h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv					S3		S2			S1			S0		
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-11	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 10-9	**S3**	**Shift 3** Specifies the multiplication factor (refer to Table 13-2. in section 13.4.).
Bits 8-6	**S2**	**Shift 2** See S3
Bits 5-3	**S1**	**Shift 1** See S3
Bits 2-0	**S0**	**Shift 0** See S3

This is a read/write register.

13.5.19. FRAME BUFFER PITCH REGISTER

This register specifies the number of bytes needed to advance from a pixel in one scan line of the Frame Buffer to the corresponding pixel in the next scan line. The value is always positive.
For example if the current screen resolution is 640x480 and the pixel depth is 3 bytes per pixel then the value programmed in this register should be 640*3 bytes.

Note:

This register is not double buffered and its value cannot be changed if the LDE is busy or the tile RAMs are dirty.

FBPITCH_REG Access: = GBASE + 4D5000h Regoffset = 0x48h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv			P												
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-13	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 12-0	**P**	**Pitch**

This is a read/write register.

13.5.20. MAXIMUM CLIP COORDINATES REGISTER

This register defines the upper clip coordinates of the rectangle boundary beyond which the LDE will not render pixels. Any coordinates beyond clip-x max and clip-y max will not be rendered. The y coordinate is in scan lines, the x coordinate in pixels per scan line.

For lower clip x/y coordinates of the rectangle refer to section 13.5.26. "Minimum Clip Coordinates Register".

CLIPMAXCORDI_REG Access: = GBASE + 4D5000h Regoffset = 0x4Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv						YUCC									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv					XUCC										
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-26	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 25-16	**YUCC**	**Y Upper Clip Coordinate**
Bits 15-11	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 10-0	**XUCC**	**X Upper Clip Coordinate**

This is a read/write register.

13.5.21. COMMAND REGISTER

Any write to this register issues a start command to the line draw engine which then begins to render the line. All the required registers must be programmed before issuing a command to LDE. Reads to this register will return zeros.

COMMAND_REG Access: = GBASE + 4D5000h Regoffset = 0x50h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
COM															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COM															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**COM**	**Command register** This can be any value.

This is a write only register.

13.5.22. STATUS REGISTER

Various bits of the 32-bit status register are used to indicate different status information as described below. Writes to this register have no effect.

STATUS_REG Access: = GBASE + 4D5000h Regoffset = 0x54h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GB	WNA	WDI	FA	TDS0	TDS1	FFT0	FFT1	Rsv							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bit 31	**GB**	**LDE Busy** ‘1’ = GE II is busy performing the programmed operation. ‘0’ = GE II is ready to start a new command.
Bit 30	**WNA**	**Write Not Allowed** ‘1’ = LDE is busy executing first operation and all the required registers and the next command are programmed for the next operation by the CPU. All the double buffered registers are in a locked state. The software can pipeline the next operation. This bit will stay ‘1’ while the LDE registers are in a locked state.
Bit 29	**WDI**	**Write Data Ignored** ‘1’ = the LDE registers are in a locked state so any writes to double buffered registers will be ignored and any reads will return the programmed data of the register. Once a pipeline command is issued there is no way to cancel that command. This is a self resetting bit and resets itself on a status register read.
Bit 28	**FA**	**Flush Active** ‘1’ = tile flush is active. The bit will be set to ‘1’ if a normal or force flush is active.
Bit 27	**TDS0**	**Tile 0 Dirty Status** ‘1’ = tile 0 is dirty, i.e. it contains some valid pixel information.
Bit 26	**TDS1**	**Tile 1 Dirty Status** ‘1’ = tile 1is dirty, i.e. it contains some valid pixel information.
Bit 25	**FFT0**	**Force Flush Tile 0** ‘1’ = the force flush for tile 0 is active.
Bit 24	**FFT1**	**Force Flush Tile 1** ‘1’ = the force flush for tile 1 is active.
Bits 23-0	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.

This is a read only register.

13.5.23. SOFTWARE RESET REGISTER

Software reset is equivalent to system reset. Writing ‘0x01h’ to this register will generate the software reset to LDE. This feature causes a reset of all internal registers, all internal FIFOs and state machines. The read value of this register is always zero.

SOFT_RESET_REG Access: = GBASE + 4D5000h Regoffset = 0x58h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SR															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SR															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**SR**	**Soft Reset Control**

This is a write only register.

13.5.24. PERFORMANCE REGISTER

This register counts the number of hclocks taken to complete a single line draw operation. Bit [5] of the Misc register must be '1' to enable performance counting (refer to section 13.5.16. "Misc Register"). The register counts the hclocks from the start of the line draw command to the end of the line draw operation. If the performance metering is enabled for the second line then the old value is over written with the performance count of the second line. Performance metering can be enabled/disabled for each line. This register is used for debugg purposes.

PERFORMANCEREG Access: = GBASE + 4D5000h Regoffset = 0x5Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PMC															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMC															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**PMC**	**Performance Metering Count**

This is a read only register.

13.5.25. FLUSH COUNT REGISTER

Tile RAMs are flushed automatically, if there are no read/write requests to the tile RAMs, after the count specified in this register. For force flushing tile RAMs refer to section 13.5.16. "Misc Register" bit [30]. If this register is set to 0 there will be no flush, if set to any other value the force flush will be set.

FLUSH_REG Access: = GBASE + 4D5000h Regoffset = 0x60h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FC															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FC															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**FC**	**Automatic tile RAM Flush Count**

This is a read/write register.

13.5.26. MINIMUM CLIP COORDINATES REGISTER

This register defines the lower clip coordinates of the rectangle boundary beyond which the LDE will not render pixels. Any coordinates beyond clip-x min and clip-y min will not be rendered. The y coordinate is in scan lines, the x coordinate in pixels per scan line.

For upper clip x/y coordinates of the rectangle refer to section 13.5.20. "Maximum Clip Coordinates Register".

CLIPMINCORDI_REG Access: = GBASE + 4D5000h Regoffset = 0x64h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv						YLCC									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv					XLCC										
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-26	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 25-16	**YLCC**	**Y Lower Clip Coordinate**
Bits 15-11	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 10-0	**XLCC**	**X Lower Clip Coordinate**

This is a read/write register.

13.6. LINE DRAWING EQUATIONS AND FUNCTIONS

For DX =0 and DY = 0 the [T] (test)

Xmajor= (abs(DX)>abs(DY))

Scale (val, max_val) = ((((long)(val)<<15)-((long)(val)<<10))/((long)(max_value)))

AA	SP	EPC	Test	Function	Values to enter
0	0	x	Yes		X0,Y0 X1, Y1 Correct = Max_correct (0x10h) Blend = 0
0	1	x	Yes	Pickpoint Fxy, Pickpoint Null,	X0, Y0, Dx X1, Y1, Dy Fxy, Dxy Correct = Max_correct (0x10h) Blend = 0
1	0	x	Yes	If (xmajor) max_dist= abs(3 dx) else abs(3dy)	X0, Y0 X1, Y1 Scale Dx = Scale (Dx, maxdistance) Scale Dy = Scale (Dy, maxdistance) Correct = Max_correct (0x10h) Blend = 0
1	1	0	Yes	Pickpoint Fxy Pickpoint Null If (xmajor) max_dist= abs(3 dx) else abs(3dy)	X0, Y0, Dx X1, Y1, Dy Fxy, Scale Dx, Scale Dy Scale Fxy = Scale (Fxy, max distance) Correct = Max_correct (0x10h) Blend = 0
1	1	1	Yes	EPC	X0, Y0, Dx X1, Y1, Dy Fxy, Scale Dx, Scale Dy Scale Fxy Correct Blend = 0

14. ALPHA FONT ENGINE

14.1. INTRODUCTION

The Alpha Font Engine (AFE) smoothes the jagged lines of enlarged fonts. This is achieved using four bits, whose value is called alpha, to represent each dot (pixel) and AFE hardware.

14.1.1. FEATURES

- Special software reset support.
- Programmable width and height for font dimension, maximum size of fonts supported is 1024x1024 pixels.
- Variable pitch.
- Nibble access capabilities for alpha values from Frame Buffer memory.
- Works with sources from both Data Port as well as screen (Frame Buffer).
- Packed as well as unpacked pixel support.
- All registers are double buffered, unless otherwise stated.
- Support for 16-bit, 24-bit and 32-bit colour format.
- Hidden source, destination and data drain FIFOs.
- Status register indicating the present status of the AFE.
- Debug command supported.
- Programmable low and high water marks for source, destination and data drain FIFOs.
- Performance monitoring register provided.

14.1.2. DOUBLE BUFFERING

The AFE can be programmed to render two lines, one after the other i.e. the software can pipeline two commands for the AFE. Once the parameters for the first line are programmed and the command is issued, all the AFE double buffered registers are available for programming and issuing the command for the next line. For further information about pipeline operation see section 14.7.8. WNA and WDI bits.

14.2. MEMORY ADDRESS SPACE

In the STPC Architecture, a 16 Mbyte memory space is allocated for extended graphics and video functions. The start of the 16 Mbyte memory space is programmable using the GBASE (in CRTC regs).

This extended graphics memory is divided into four 4 MByte regions, as shown in Figure 14-1. The second 4 MBytes (Region 1) is used for memory mapped registers needed for extended graphics and video functionality. This 4 MByte region is further subdivided into sixteen 256 KByte regions. The fourth 256 KByte (Region 3) is further subdivided into four 64 KBytes and each of these 64 KBytes is subdivided into sixteen 4 KBytes. The Alpha Font Engine (AFE) uses the fifth 4 KByte region for its register access

Access to non-implemented register space will be ignored, and cycles will be terminated normally.

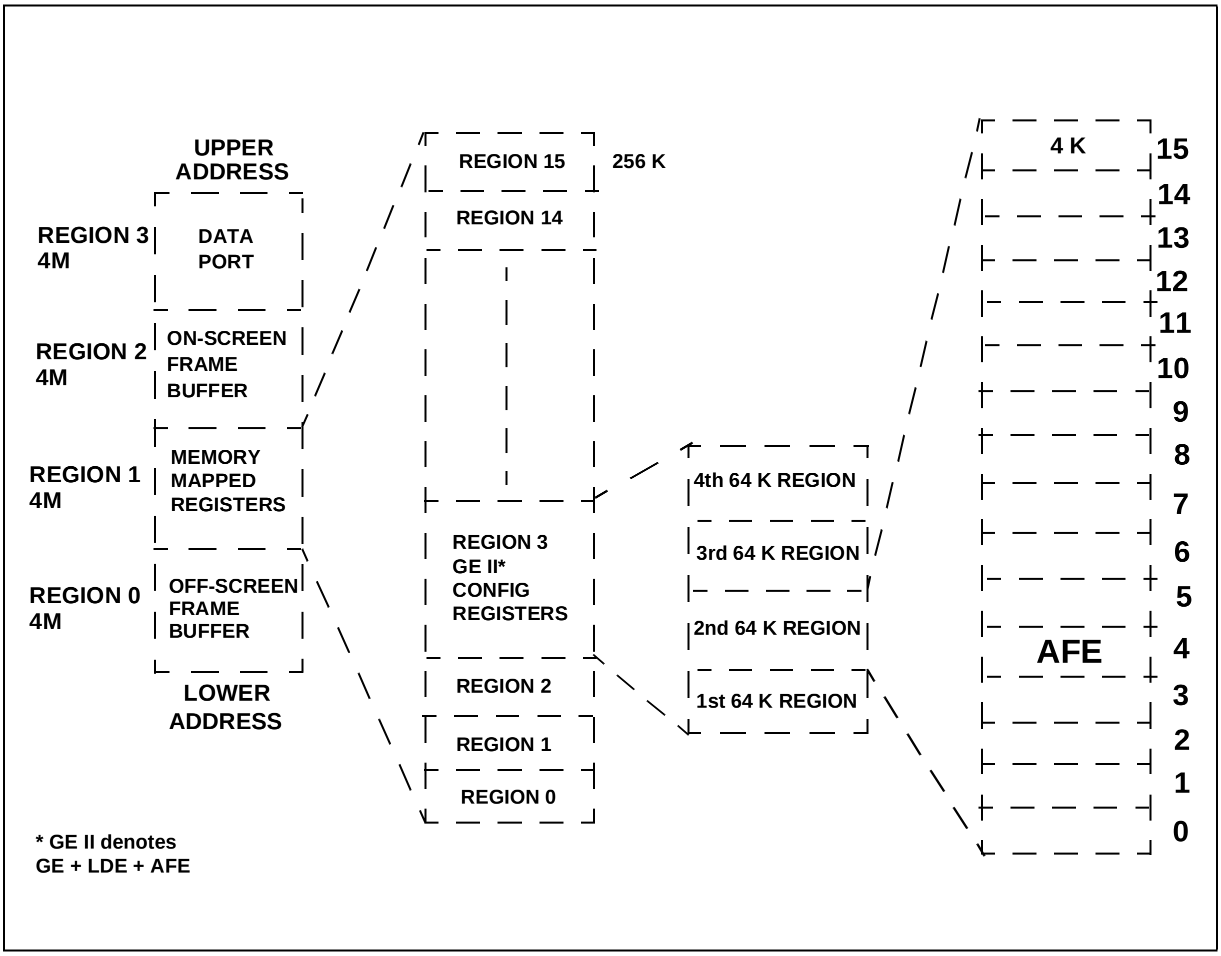


Figure 14-1. Extended Graphics Memory Map

14.3. REGISTER ACCESS

All the registers are accessible via the memory mapped 4 MByte Region 1 in the High Linear Frame Buffer.

The Font, Dst_XY and Dst_Base registers can also be programmed during the command initialization process:

00001	**<GBASE>**	**01**	**0011**	**01**	**0100**	**<REG>**
31 27	26 24	23 22	21 18	17 16	15 12	11 0
		4 MByte Region 1 (memory mapped region)	256 KByte Region 3	2nd 64 KByte Region	5th 4 KByte Region	
Default value after reset = GBASE + 4D4000h						

Table 14-1. Register Access

<REG> specifies the offset of the register to be accessed. Read may be performed for any width, but write must be a 32-bit transfer.

Configuration registers are double buffered with the following **exceptions:**
Misc_Reg and Soft_Reset_Reg

The CPU can program all the registers and can issue a command. Once the status bit [31] has a value ‘1’, indicating “AFE_Busy”, all the registers can be programmed for the next command. The CPU can issue another command in the pipeline. At this point, since the first command is not yet over, the status bit [30] has a value ‘1’, indicating “no more writes to the registers is allowed” (since both the buffers are full). Then if the CPU tries to update any register, the cycle is terminated properly but the write data is ignored and the status bit [29] has a value ‘1’, indicating “overwrite data ignored”. Reads to any register can be made at any time.

14.4. ALPHA FONT ENGINE OPERATION

To perform alpha font rendering, program the registers as follows (the sequence of programming is not important):

1) Program the destination base address of the screen or region into Dst_base.

2) Program the destination XY register.

3) Program the foreground colour register.

4) Program the source base address (only if the source is Frame Buffer).

5) Program the source pitch (only if the source is Frame Buffer).

6) Program the source XY (only if the source is Frame Buffer).

7) Program the destination pitch.

8) Program the pixel depth for number of bytes per pixel.

9) Program the height and width of the font.

10) Program the background colour (only if the destination is background colour register).

11) Program the ROP code in ROP code register if required.

12) Octave register (only if the source or the destination FIFO fill will access Frame Buffer).

13) Finally write to the command register through the command port to initiate the alpha font operation.

Some of the parameters, such as destination pitch, background colour and pixel depth, will already be set up from prior operations and will not need to be written each time.

14.5. OPERAND FRAME BUFFER ADDRESSES

AFE fetches the needed data from the Frame Buffer area. The software identifies these areas with an operand base address, unsigned X and Y offsets from this base address and a pitch for that region. Pitch is the byte distance between two pixels in the same X position of adjacent scan lines.

Frame buffer is addressed using SDRAM linear addresses. These are the addresses that the SDRAM is presented with. The Frame Buffer starts at SDRAM linear address '0' and continues to the top of the Frame Buffer. The system physical addresses are mapped to above the Frame Buffer. To accommodate the more natural view of the Frame Buffer, AFE implements XY addressing. An operand's base address, pitch, X and Y components are combined in AFE to form the associated SDRAM linear address. The base component of the operand is the SDRAM linear address of the start of that operand. The address can range from '0' to the maximum size of the Frame Buffer, depending upon where the operand is located in the Frame Buffer.

Internally, AFE performs its calculations using the XY coordinates. When an SDRAM address is needed, for example to write a destination pixel, the address is computed using:

Linear_address = operand base + X + (Y * pitch)

The multiplication of the pitch is done by hardware shifts and add. The pitch value is specified as a group of four shift codes. For each non-zero shift code the Y address is shifted by the corresponding number of bits and then added. The shift values supported are:

Value	dstpitch[10:9] shift 3	dstpitch[8:6] shift 2	dstpitch[5:3] shift 1	dstpitch[2:0] shift 0
000	0	0	0	0
001	1024 * dstY	32 * dstY	0	0
010	2048 * dstY	64 * dstY	64 * dstY	0
011	4096 * dstY	128 * dstY	128 * dstY	128 * dstY
100	n/a	256 * dstY	256 * dstY	256 * dstY
101	n/a	512 * dstY	512 * dstY	512 * dstY
110	n/a	0	1024 * dstY	1024 * dstY
111	n/a	0	0	2048 * dstY

Table 14-2. Shift Values Supported

The operand base addresses must be aligned to 32 bytes (i.e. the five least significant bits of the address must be zeros). The supported pitches (in bytes) are:

0	544	1216	1792	2432	3136	4160	4704	5408	6432
32	576	1248	1824	2464	3168	4192	4736	5440	6464
64	608	1280	1856	2496	3200	4224	4768	5504	6528
96	640	1312	1920	2560	3232	4256	4800	5632	6656
128	672	1344	2048	2592	3264	4288	4864	5664	6688
160	704	1376	2080	2624	3328	4320	4896	5696	6720
192	768	1408	2112	2656	3360	4352	4928	5760	6784
224	800	1440	2144	2688	3392	4384	4992	5888	6912
256	832	1472	2176	2720	3456	4416	5120	6144	7168
288	896	1536	2208	2752	3584	4448	5152	6176	7200
320	1024	1568	2240	2816	3616	4480	5184	6208	7232
352	1056	1600	2272	2848	3648	4512	5216	6240	7296
384	1088	1632	2304	2880	3712	4544	5248	6272	7424
416	1120	1664	2336	2944	3840	4608	5280	6304	7680
448	1152	1696	2368	3072	4096	4640	5312	6336	
512	1184	1728	2400	3104	4128	4672	5376	6400	

Table 14-3. Pitches Supported

14.6. COMMAND INITIATION

Make sure that all other relevant registers are programmed before issuing a command register write.

The command register appears several times in the register address space and writing to different addresses has different effects. The writing address is controlled by the cmd offset. Writing to the command register address causes the alpha font operation of the command selected by the cmd offset to start (except for the debug command).

00001	<GBASE>	01	0011	01	0100	<cmd>
31 27	26 24	23 22	21 18	17 16	15 12	11 0
		4 MByte Region 1 (memory mapped region)	256 KByte Region 3	2nd 64 KByte Region	5th Region	
Default value after reset = GBASE + 4D4000h+ <CMD>						

Table 14-4. Address Bit Decode

Bits [23:12] (01_0011_01010_0) identify this as a command and register initiation register access. Cmd offsets and their effects are:

CmdOffset	cmdName	Decoded as
0x50h	cmd Font	**Operation mode.** The data in data phase will be written to **Font Register** and the operation starts, write only register.
0x54h	cmd DstXY	**Operation mode.** The data in the data phase will be written in **Dst_XY** register and the operation starts, write only register.
0x58h	cmd DstBase	**Operation mode.** The data in the data phase will be written in **Dst_base** register and the operation starts, write only register.
0x5Ch	cmd Debug	**Debug mode**. Writes the data in the data phase to the font register without starting the alpha font rendering operation, write only register.

Table 14-5. Decode of cmd Bit

Notes:

1) The Font, Dst_XY and Dst_Base are also programmable through the normal register access mechanism. For details refer to section 14.7. "Register Description".

2) Access to non-implemented command/register address space will be ignored, and the cycle will be terminated normally.

14.7. REGISTER DESCRIPTION

All registers (except Misc and Soft_Reset) are double buffered. If there is a specific consideration for any register then it is mentioned along with the explanation of the register. The following applies to all double buffered registers:

At any given point AFE can perform one operation. However, after issuing the first command, the **double buffered** registers can be programmed for the next operation and the next command can also be issued. At this point all the double buffered registers enter a locked state i.e. that the first command is still in progress and the next command has also been issued. In this state if any writes are made to any of the double buffered registers, then the CPU cycle is terminated normally and the write data is ignored. Registers can however be read at any time.

14.7.1. BACKGROUND COLOUR REGISTER

This register contains the full-colour value(s). It only needs to be programmed if the destination operand type in the ‘ROP register’ (section 14.7.12.), for this command, is ‘background register’ (setting ‘ROP register’ bit [12] to ‘1’).

For 16-bit pixels, only bits 15-0 are significant.
For 24-bit pixels, only bits 23-0 are significant.

BACKGROUND Access = GBASE + 4D4000h Regoffset = 0x00h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BC															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BC															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	BC	**Background colour** This is the colour to be used as the background.

Programming note:

The content of this register is not altered by rendering operations.

14.7.2. DATA PORT REGISTER

The CPU can write to the Data Port at any time. There is a hidden 16 quad-word deep FIFO. If the Alpha Font Engine is unable to accept any additional writes to the Data Port, the CPU will hold the data until the controller is ready to accept more data in the Data Port FIFO. The low water mark of the Data Port FIFO (source FIFO) is programmable and the status register (section 14.7.8.) uses bit [28] to indicate the status of the Data Port FIFO. This status bit indicates if the Data Port FIFO is below the low water mark. Initially the CPU must make a 64-bit write (two 32-bit writes) to the FIFO. It can then poll the low water mark status bit to make more writes.

If the last write only needs to be 32 bits (dpOdd, status register bit [18], will be ‘1’, refer to section 14.7.8.) the CPU must make one extra (32-bit) write to Data Port.

Writing to the Data Port address is the same as writing to any double word between
(128 MByte+(GBASE << bit 24) + 12 MByte) and (128 MByte+(GBASE << 24) + 16 MByte).

Data port’s 4 MByte alias is common to GE I and GE II, where GE I refers to Graphic Engine (GE) and GE II refers to Graphic Engine plus Alpha Font Engine plus Line Drawing Engine (GE + AFE + LDE). Data port writes are directed to GE I or GE II according to the setting of misc register (section 14.7.17.) bit [0].
Bit [0] = 0 (default) writes directed to GE I.
Bit [0] = 1 writes directed to GE II.

DATA PORT　　Access = GBASE + 4D4000h　　Regoffset = 0x04h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DP															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DP															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**DP**	**Data Port**

Note:

If the source for alpha values is Frame Buffer (refer to section 14.7.12. "ROP" bit [12] destination operand type) then a write to Data Port will be ignored, the dpOdd (bit [18] in status register) will not be affected, the write not allowed (bit [30] in status register) will be set and the write cycle will be terminated normally. For more details refer to section 14.7.8. "Status Register".

Packet Data

*					
	Packet 1	Packet 2	Packet 3	Packet 4	Packet X

* Number of Alpha values ignored = bit [2:0] of the

Unpacket Data

*			Addoffset				
Packet 1	**Packet 2**	**Packet 3**	**Packet 4**	**Packet 5**	**Packet 6**		

* Number of Alpha values ignored = bit [2:0] of the
For addoffset see section 14.7.16.

14.7.3. DESTINATION BASE ADDRESS REGISTER

This register specifies the starting SDRAM linear physical address of the destination operand (aligned to a 32 byte boundary).

Note:

This register is also accessible for the write operation through the command address space, refer to section 14.6. "Command Initiation" for more details.

DST_BASE Access = GBASE + 4D4000h Regoffset = 0x08h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DB															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DB															
Default value after reset = 0x00h															

Bit Number	Mnemonic	Description
Bits 31-0	DB	Base SDRAM linear address of the destination operand. The lower 5 significant bits are always ‘0’.

Programming note:

The content of this register is not altered by rendering operations.

14.7.4. DESTINATION PITCH REGISTER

This register specifies the number of bytes needed to advance from a pixel in one scan line of the destination to the corresponding pixel in the next scan line. The pitch value is specified as the number of bytes in one scan line and its value is always positive.

Pitch calculation:
If there are 640 pixels in a scan line and each pixel is represented by 4bits,
then pitch = (640 x 4)/8 = 320 bytes.
Similarly if 1bit/pixel is used then pitch = (640 x 1)/8 = 80 bytes.

DST_PITCH　　Access = GBASE + 4D4000h　　Regoffset = 0x0Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv					DS3		DS2			DS1			DS0		
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-11	Rsv	**Reserved** - Read will return ‘0’s for these bits.
Bits 10-9	DS3	**Dst_shift3** Specifies an amount by which to multiply Dst_XY.Y (section 14.7.5.). The result, along with the other shift results, is added to the Dst_base (section 14.7.3.) and Dst_XY.X to compute the SDRAM linear address of the destination pixel. For further details of SDRAM linear addresses refer to section 14.5. "Operand Frame Buffer addresses".
Bits 8-6	DS2	**Dst_shift2** See Dst_shift3.
Bits 5-3	DS1	**Dst_shift1** See Dst_shift3.
Bits 2-0	DS0	**Dst_shift0** See Dst_shift3.

Refer to Table 14-2 "Shift Values Supported" in section 14.5.

Programming note:

The content of this register is not altered by rendering operations.

14.7.5. DESTINATION COORDINATES REGISTER

This register contains the co-ordinate address of the starting corner of the destination operand.

This register is also accessible for the write operation through the command address space, refer to section 14.6. "Command Initiation" for more details.

DST_XY Access = GBASE + 4D4000h Regoffset = 0x10h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DY															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DX															
Default value after reset = 0x00h															

Bit Number	Mnemonic	Description
Bits 31-16	**DY**	The unsigned Y-coordinate of the starting corner of the destination operand.
Bits 15-0	**DX**	The unsigned X-coordinate of the starting corner of the destination operand.

Programming note:

The content of this register is not altered by rendering operations.

14.7.6. FOREGROUND COLOUR REGISTER

For 16-bit pixels, only bits [15:0] are significant.
For 24-bit pixels, only bits [23:0] are significant.

FOREGROUND   Access = GBASE + 4D4000h   Regoffset = 0x14h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FCI															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FCI															
Default value after reset = 0xFFFFFFFFh															

Bit Number	Mnemonic	Description
Bits 31-0	**FCI**	This is the colour to be used as the foreground colour.

Programming note:

The content of this register is not altered by rendering operations.

14.7.7. FONT REGISTER

This register defines the height and width of the current font to be blended.

This register is also accessible for the write operation through the command address space, refer to section 14.6. "Command Initiation" for more details.

FONT Access = GBASE + 4D4000h Regoffset = 0x18h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv						W									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv						H									
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-26	Rsv	**Reserved** - Read will return ‘0’s for these bits.
Bits 25-16	W	**Width** Specifies one less than the actual pixels in one scan line.
Bits 15-10	Rsv	**Reserved** - Read will return ‘0’s for these bits.
Bits 9-0	H	**Height** Specifies one less than the number of scan lines in the source Frame Buffer region.

Programming note:

The content of this register is not altered by rendering operations.

14.7.8. STATUS REGISTER

This register provides AFE status information.

STATUS — Access = GBASE + 4D4000h — Regoffset = 0x1Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GB	WNA	WDI	SWM	DWM	DDWM	ABPS	SF	SE	DF	DE	DDF	DDE	dpOdd	Rsv	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv															
Default value after reset = 0x18A80000h															

Bit Number	Mnemonic	Description
Bit 31	**GB**	**AFE Busy** Indicates the busy status of the engine. ‘1’ = AFE is busy performing the programmed operation. ‘0’ = AFE is ready to start the new command.
Bit 30	**WNA**	**Write Not Allowed** ‘1’ = AFE is busy executing the first operation and all the required registers and the next command are programmed for the next operation by the CPU. All the double buffered registers are in a locked state. The software can pipeline the next operation. This bit will keep the value ‘1’ while the AFE registers are in a locked state.
Bit 29	**WDI**	**Write Data Ignored** ‘1’ = the AFE registers are in a locked state so any writes to double buffered registers will be ignored and any reads will return the programmed data of the register. Once a pipeline command is issued there is no way to cancel that command. This is a self resetting bit and resets itself on a status register read. This bit also gets set when source of alpha values is Frame Buffer (ROP (section 14.7.12.) bit [11] is ‘0’) and there is write access to Data Port. This write to Data Port will be ignored but status bit [18] (dpOdd) will not be affected.
Bit 28	**SWM**	**Source FIFO below Water Mark** The source FIFO is common for the alpha font data coming from either the Data Port or the Frame Buffer (refer to section 14.7.7. "Font Register" bit [23]). The water mark can be programmed (refer to section 14.7.13. "Source Water Mark Register"). ‘1’ = the source FIFO is below the water mark (low priority read).
Bit 27	**DWM**	**Destination FIFO below Water Mark** The source FIFO is common for destination data coming from either the ‘Background’ register or the Frame Buffer (refer to section 14.7.7. "Font Register" bit [24]). The water mark can be programmed (refer to section 14.7.14. "Destination Water Mark Register"). When the ‘Background’ register is used as destination data then this status bit is meaningless. ‘1’ = the destination FIFO is below the water mark (low priority read).

Bit 26	**DDWM**	**Data Drain FIFO above Water Mark** The data drain FIFO is used to store the processed data to be written to the destination Frame Buffer. The water mark for this FIFO can be programmed (refer to section 14.7.15. "Data Drain Water Mark Register"). '1' = the data drain FIFO is above the water mark (low priority write).
Bit 25	**ABPS**	**Alpha Blend Pipeline Stall** '1' = the alpha blend pipeline is in stall state. The reason for the stall can then be established from bits [28:26] above.
Bit 24	**SF**	**Source FIFO Full** '1' = full, used only for debugging
Bit 23	**SE**	**Source FIFO Empty** '1' = empty, used only for debugging
Bit 22	**DF**	**Destination FIFO Full** '1' = full, used only for debugging
Bit 21	**DE**	**Destination FIFO Empty** '1' = empty, used only for debugging
Bit 20	**DDF**	**Data Drain FIFO Full** '1' = full, used only for debugging
Bit 19	**DDE**	**Data Drain FIFO Empty** '1' = empty, used only for debugging
Bit 18	**dpOdd**	**Data Port Odd Write** CPU accesses are 32 bit aligned and source FIFO is 64 bit aligned. When the CPU writes the first 32 bit data value, the dpOdd bit is set to '1' by Data Port to indicate to the CPU that to complete 64 bit FIFO write it requires one more 32 bit write access. It is essential to check this bit on the last access for Data Port in the current alpha font processing command. When the source of alpha values is Frame Buffer (refer to section 14.7.12. "ROP" bit [11]) and there is write access to dataport, this write to dataport will be ignored and dpOdd will not be affected.
Bits 17-0	**Rsv**	**Reserved** - Read will return '0's for these bits.

14.7.9. SOURCE COORDINATES REGISTER

This register contains the coordinate address of the top left starting corner of the source operand.

The alpha values coming from source to alpha font may be neither 32-bit (Data Port writes) nor 64-bit (Frame Buffer access) aligned. Since the source FIFO is 64-bit wide, the starting alpha nibble can be located at any of the 16 possible nibble positions. Bits [3:0] are used to point to the starting nibble for the first pixel processing in every scan line. It points to the 4 bit alpha value of the pixel.

Note:

There is no direction control when rendering an alpha font.

SRC_XY Access = GBASE + 4D4000h Regoffset = 0x20h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SY															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SX															
Default value after reset = 0x00h															

Bit Number	Mnemonic	Description
Bits 31-16	**SY**	The unsigned Y-coordinate of the starting corner of the source operand.
Bits 15-0	**SX**	The unsigned X-coordinate of the starting corner of the source operand. The lower three significant bit correspond to the packed data shift.

Programming note:

The content of this register is not altered by rendering operations.

14.7.10. SOURCE BASE ADDRESS REGISTER

This register specifies the starting SDRAM linear address of the source operand.

For an unpacked alpha value fetch from Frame Buffer program bits [21:05] as actual base (32byte aligned) and bits [4:3] as 00. Also program Src_Pitch (section 14.7.11.) and Src_XY (section 14.7.9.) for complete address calculation, as the address will be calculated by AFE.

For a packed alpha value fetch from Frame Buffer program all bits i.e. bits [21:03], as the total address will be calculated by the CPU. Src_XY should be programmed only for providing nibble access position.

SRC_BASE　　Access = GBASE + 4D4000h　　Regoffset = 0x24h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SB															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SB															
Default value after reset = 0x00000h															

Bit Number	Mnemonic	Description
Bits 31-0	**SB**	**Source Base** Base SDRAM linear address of the source operand. The lower three significant bit correspond to the packed data shift.

Programming note:

The content of this register is not altered by rendering operations.

14.7.11. SOURCE PITCH REGISTER

This register specifies the number of bytes needed to advance from a pixel in one scan line of the source to the corresponding pixel in the next scan line. This value is always positive.

SRC_PITCH Access = GBASE + 4D4000h Regoffset = 0x28h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv					SS3		SS2			SS1			SS0		
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-11	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 10-9	**SS3**	**Src_shift3** Specifies an amount by which to multiply Src_XY.Y (section 14.7.9.). The result, along with the other shift results, is added to the Dst_base (section 14.7.3.) and Src_XY.X to compute the SDRAM linear address of the destination pixel. For further details of SDRAM linear addresses refer to section 14.5. "Operand Frame Buffer addresses".
Bits 8-6	**SS2**	**Src_shift2** See SS3.
Bits 5-3	**SS1**	**Src_shift1** See SS3.
Bits 2-0	**SS0**	**Src_shift0** See SS3.

Programming note:

The content of this register is not altered by rendering operations.

14.7.12. ROP

ROP Access = GBASE + 4D4000h Regoffset = 0x2Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv			D	S	PS	PD		Rsv				R			
Default value after reset = 0x0000000Ch															

Bit Number	Mnemonic	Description
Bits 31-13	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bit 12	**D**	**Destination operand type** ‘1’ = the destination operand is background register. ‘0’ = the destination operand is Frame Buffer. In this case the background register does not need to be programmed.
Bit 11	**S**	**Source Operand type** ‘1’ = the source operand is Data Port. In this case the Src_XY, Src_Base and Src_Pitch registers need not be programmed. ‘0’ = the source operand is Frame Buffer. When this bit is ‘0’, if a write to the Data Port is made, the CPU cycle will be terminated normally and the write will be ignored.
Bit 10	**PS**	**Packed Source data** ‘0’ = the font is in unpacked format. The unused bits of the last 32 bits sent to the Data Port are discarded by AFE + Addoffset. For next line (on screen) new 32 bits of data are used. ‘1’ = the font is in packed format. The unused bits of the last 32 bits sent to the Data Port are used as the beginning of the next line. If Frame Buffer is the source of the packed data then pitch will not be used for calculating the pixel coordinates for the beginning of the next line. A packed format is one where the source alpha values in the Frame Buffer are stored in sequential SDRAM addresses, row after row. Specifying the pitch equal to the width number of pixels also has the effect of packed source data.
Bits 9-8	**PD**	**Pixel Depth** Specifies the number of bits per pixel. 0 = not defined. 1 = 2 bytes per pixel. 2 = 3 bytes per pixel. 3 = 4 bytes per pixel. For detailed information about the colour expansion method refer to section 14.7.17. "MISC Register" bit [1].
Bits 7-4	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 3-0	**R**	**ROP Code** This code enables the font colour to be generated from different combinations of foreground colour register value and destination colour (either destination colour register or from destination Frame Buffer). The codes and their functions are listed in Table 14-6.

ROP CODE	ROP	DESCRIPTION OF FONT COLOUR
0x0h	R2_BLACK (font Black)	always ‘0’
0x1h	R2_NOT_MERGE_FONT	inverse of R2_MERGE_FONT
0x2h	R2_MASK_NOT_FONT	combination of colours common to destination colour and inverse of foreground colour register value
0x3h	R2_NOT_COPY_FONT (font invert)	inverse of foreground colour register value
0x4h	R2_MASK_FONT_NOT	combination of colours common to both foreground colour register value and inverse of destination colour
0x5h	R2_NOT	inverse of destination colour
0X6h	R2_XOR_FONT	combination of colours in the foreground colour register value and destination colour, but not in both
0x7h	R2_NOT_MASK_FONT	inverse of R2_MASK_FONT
0x8h	R2_MASK_FONT	combination of colours common to both foreground colour register value and destination colour
0x9h	R2_NOT_XOR_FONT	inverse of R2_XOR_FONT colour
0xAh	R2_NOP (font erase)	remains unchanged (i.e. font colour is destination colour only)
0xBh	R2_MERGE_NOT_FONT	combination of the destination colour and inverse of foreground colour register value
0xCh	R2_COPY_FONT	foreground colour register value
0xDh	R2_MERGE_FONT_NOT	combination of foreground colour and inverse of destination colour
0xEh	R2_MERGE_FONT	combination of foreground colour and destination colour
0xFh	R2_WHITE(font white)	always white

Table 14-6. Possible ROP Codes and their Significance

14.7.13. SOURCE WATER MARK REGISTER

The water mark indicates the **emptiness of the FIFO**. The source FIFO in the AFE is 16 quad words deep and the three water marks can point to any of the 0-15 quad word locations in the order:
water mark 1 < water mark 2 < water mark 3.
The water mark is used by the Frame Buffer access arbitration logic to calculate the priority of source FIFO read over destination FIFO read and data drain FIFO write as follows:

Source FIFO contents	**Priority for memory access**
>= water mark 3	11 (highest)
>= water mark 2	10
>= water mark 1	01
< water mark 1	00 (lowest)

For example, if water mark 1 points to quad word 4 then:
when the FIFO is less than location 4 full, status register bit [28] (section 14.7.8.) will be ‘1’,
when the FIFO contents are at level 4 or above, bit [28] will be ‘0’.

SRC_WATER_MARK Access = GBASE + 4D4000h Regoffset = 0x30h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv												SWM3			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv				SWM2				Rsv				SWM1			
Default value after reset = 0x000C0804h															

Bit Number	Mnemonic	Description
Bits 31-20	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 19-16	**SWM3**	**Source FIFO water mark 3**
Bits 15-12	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 11-8	**SWM2**	**Source FIFO water mark 2**
Bits 7-4	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 3-0	**SWM1**	**Source FIFO water mark 1**

14.7.14. DESTINATION WATER MARK REGISTER

The water mark indicates the **emptiness of the FIFO**. The destination FIFO in the AFE is 16 quad words deep and the three water marks can point to any of the 0-15 quad word locations in the order:
water mark 1 < water mark 2 < water mark 3.
The water mark is used by the Frame Buffer access arbitration logic to calculate the priority of destination FIFO read over source FIFO read and data drain FIFO write as follows:

Destination FIFO contents	Priority for memory access
>= water mark 3	11 (highest)
>= water mark 2	10
>= water mark 1	01
< water mark 1	00 (lowest)

For example, if water mark 1 points to quad word 4 then:
when the FIFO is less than location 4 full, status register bit [28] (section 14.7.8.) will be ‘1’,
when the FIFO contents are at level 4 or above, bit [28] will be ‘0’.

DST_WATER_MARK   Access = GBASE + 4D4000h   Regoffset = 0x34h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv												DWM3			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv				DWM2				Rsv				DWM1			
Default value after reset = 0x000C0804h															

Bit Number	Mnemonic	Description
Bits 31-20	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 19-16	**DWM3**	**Destination FIFO water mark 3**
Bits 15-12	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 11-8	**DWM2**	**Destination FIFO water mark 2**
Bits 7-4	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 3-0	**DWM1**	**Destination FIFO water mark 1**

14.7.15. DATA DRAIN WATER MARK REGISTER

The water mark indicates the **fullness of the FIFO**. The data drain FIFO in the AFE is 16 quad words deep and the three water marks can point to any of the 0-15 quad word locations in the order:
water mark 1 < water mark 2 < water mark 3.
The water mark is used by the Frame Buffer access arbitration logic to calculate the priority of destination FIFO read over source FIFO read and data drain FIFO write as follows:

Data drain FIFO contents	Priority for memory access
< water mark 1	11 (highest)
>= water mark 1	10
>= water mark 2	01
>= water mark 3	00 (lowest)

For example, if water mark 1 points to quad word 4 then:
when the FIFO is less than location 4 full, status register bit [28] (section 14.7.8.) will be ‘1’,
when the FIFO contents are at level 4 or above, bit [28] will be ‘0’.

DST_WATER_MARK   Access = GBASE + 4D4000h   Regoffset = 0x38h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv												DDWM3			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv				DDWM2				Rsv				DDWM1			
Default value after reset = 0x000C0804h															

Bit Number	Mnemonic	Description
Bits 31-20	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 19-16	**DDWM3**	**Data Drain FIFO water mark 3**
Bits 15-12	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 11-8	**DDWM2**	**Data Drain FIFO water mark 2**
Bits 7-4	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 3-0	**DDWM1**	**Data Drain FIFO water mark 1**

14.7.16. ADD OFFSET REGISTER

Add offset is used if source for alpha values is Data Port. When the width counter expires (current line end) the next nibble offset in the current quad word (beginning of next scan line) is calculated by adding this parameter to the present nibble offset.

A value must be set in this register for every command if the source operand type is ‘1’ (refer to section 14.7.12. "ROP" bit [11]).

ADD_OFFSET Access = GBASE + 4D4000h Regoffset = 0x3Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv												A0			
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-4	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 3-0	**A0**	**Add Offset -** For Mod 8 = 0 then Addofset + 1 = A0 For Mod 8 ≠ 0 then 9 - (Addoffset mod 8)= 0

14.7.17. MISC REGISTER

This register is not double buffered, hence access to this register is permitted at any time, independent of the AFE busy/write not allowed status.

MISC Access = GBASE + 4D4000h Regoffset = 0x40h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv														EXP	DPA
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-2	Rsv	**Reserved** - Read will return ‘0’s for these bits.
Bit 1	EXP	**Expand RGB** This bit is used to expand the 16-bit RGB (Pixel depth = 1, i.e. 2 bytes per pixel) into 24-bit RGB mode for processing in AFE. The expansion of 5:6:5 RGB colour format into 8:8:8 RGB colour format is shown below.
Bit 0	DPA	**Data Port Access Control** ‘1’ = all the write accesses will go to GE II Data Port ‘0’ = all the write accesses will go to GE I Data Port

5 to 8 Bits Colour Conversion

	Colour Bit No.							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Original 5 bit colour data				4	3	2	1	0
Expanded data when Misc [1] = ‘0’	4	3	2	1	0	4	3	2
Expanded data when Misc [1] = ‘1’	4	3	2	1	0	4	4	4

6 to 8 Bits Colour Conversion

	Colour Bit No.							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Original 6 bit colour data			5	4	3	2	1	0
Expanded data when Misc [1] = ‘0’	5	4	3	2	1	0	5	4
Expanded data when Misc [1] = ‘1’	5	4	3	2	1	0	5	5

14.7.18. SOFTWARE RESET REGISTER

Software reset is equivalent to system reset. By writing ‘0x01h’ to this register the user will generate the software reset to AFE. This feature causes a reset of all AFE registers, all internal FIFOs and state machines. The read value of this register is always ‘0’. The soft reset is active till 8 hclk in hclk domain and 8mclk in mclk domain. During this period any read/write to AFE will cause unpredictable results.

SOFT_RESET Access = GBASE + 4D4000h Regoffset = 0x44h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
soft reset control															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
soft reset control															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	**SR**	**Soft Reset Control**

14.7.19. CURRENT PERFORMANCE COUNT REGISTER

This register holds the number of clock units required for the current antialiased font drawing of the given size. The number is the count of mclks.

CUR_PERF_REG Access = GBASE + 4D4000h Regoffset = 0x48h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv												CP			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CP															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-20	**Rsv**	**Reserved** - Read will return ‘0’s for these bits.
Bits 19-0	**CP**	**Current Performance Count**

14.7.20. PREVIOUS PERFORMANCE COUNT REGISTER

This register holds the number of clock units required for the previous antialiased font drawing of the given size. The number is the count of mclks. It is automatically updated when a new command for AFE is given. This is a useful feature for comparing different operations.

PVS_PERF_REG　　Access = GBASE + 4D4000h　　Regoffset = 0x4Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv												PP			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PP															
Default value after reset = 0x00000000h															

Bit Number	Mnemonic	Description
Bits 31-20	**Rsv**	**Reserved -** Read will return ‘0’s for these bits.
Bits 19-0	**PP**	**Previous Performance Count**

14.7.21. COMMAND FONT REGISTER

This command causes the alpha font operation to start and the data to be written to the font register (refer to section 14.7.7. "Font Register"). See also section 14.6."Command Initiation".

CMD_FONT_REG Access = GBASE + 4D4000h Regoffset = 0x50h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv						W									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv						H									
Default value after reset = 0x00000000h															

14.7.22. COMMAND DST_XY REGISTER

This command causes the alpha font operation to start and the data to be written to the address in Dst_XY (refer to section 14.7.5. "Destination Coordinates Register"). See also section 14.6."Command Initiation".

CMD_DSTXY_REG Access = GBASE + 4D4000h Regoffset = 0x54h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DY															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DX															
Default value after reset = 0x00h															

14.7.23. COMMAND DST_BASE REGISTER

This command causes the alpha font operation to start and the data to be written to the address in Dst_Base (refer to section 14.7.3. "Destination Base Address Register"). See also section 14.6."Command Initiation".

CMD_DSTBASE_REG Access = GBASE +4D4000h Regoffset = 0x58h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DB															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DB															
Default value after reset = 0x00h															

14.7.24. COMMAND DEBUG REGISTER

This command writes to the address in the font register (refer to section 14.7.7. "Font Register") without starting the alpha font operation. See also section 14.6."Command Initiation".

CMD_DEBUG_REG	Access = GBASE + 4D4000h	Regoffset = 0x5Ch

15. VIDEO INPUT PORT

15.1. INTRODUCTION

The STPC video input signals and buffering are managed by the Video Controller. The video input signal and buffering are controlled by the Video Input Port, and is described in the remainder of this Chapter.

The video input and display signals are controlled through the Video Pipeline registers. These registers, which control the settings for the display buffer areas, plus filter control, colour mixing and colour space mixing, are described in the separate Video Pipeline Registers Chapter .

15.2. VIDEO INPUT PORT (VIP) OVERVIEW

The purpose of the Video Input Port is to accept an encoded digital video signal stream in one of a number of industry standard formats, decode it, optionally decimate it 2:1, and deposit it into an offscreen area of the Frame Buffer. An interrupt request can be generated when an entire field or frame has been captured.

The Video Input Port includes a fully functional VIP Host Master Port with hardware polling, programmable time-out period and programmable time-slice arbitration logic. This interface implements the full VIP Host Port Protocol - burst mode, master or slave-terminated transfers, wait-states and time-out transfers.

The channel has a 32-byte FIFO to optimise transfers to system memory while ensuring adequate bandwidth for VIP transfers. Both channels support Hardware Polling with a programmable delay period to reduce polling when the selected target FIFO is not ready. Hardware polling minimises the transfer start-up time for the DMA operations.

Arbitration between DMA Channels and host accesses can be round-robin or priority based. Round-robin arbitration and maximum burst length controls allow the maximum latency to be calculated and controlled. Priority based arbitration insures maximum bandwidth for critical tasks. The time-out period is programmable to accommodate devices with long access times.

15.3. DIGITAL VIDEO INPUT FORMATS

The video input port can be programmed to decode one of several video formats. The following sections discuss this functionality in more detail.

Existing video input formats are shown in Table 15-1.

Table 15-1. Video Signal Formats

Input Format	Description
VIP 1.0 (ITU-R 656)	Lock internal timing generator to EAV codes
UTIR-601	8-bit multiplexed CCIR 601

15.3.1. VIP 1.0 COMPATIBLE VIDEO

The Video Input Port supports the simplified SAV (Start of Active Video) and EAV (End of Active Video) codes, as defined in the VIP 1.0 Specification.

In this mode, the Video Timing Generator cannot be used to specify the horizontal or vertical active periods. The capture of video data must be based solely on the SAV and EAV codes embedded in the video stream.

This implementation includes:

- The Video Timing Generator used independently to generate the system timing signals HSYNC# and B/T#.
- Horizontal and vertical active window, based on SAV and EAV codes only.
- Byte swapping may be disabled, based on the Task Bit from the SAV code.
- Invalid pixel detection when the value is 0x00. When a pixel data value of 0x00 is encountered during an active line, the data is not written to the Frame Buffer and the pointer is not incremented. This allows re-sampled video to be output without changing the PIXCLK frequency.

15.3.2. 8-BIT MULTIPLEXED ITU-R 601

This mode provides a glueless video interface to the STi3520A MPEG-2 decoder chip. The video data interface consists of eight data pins, two control pins and a pixel clock. The UTIR-601 outputs video data in 4:2:2 format, multiplexed to 8-bit data words in Cb, Y, Cr, Y format. The UTIR-601 uses input signals field (B/T#) and horizontal sync (HSYNC#) to generate video timing. The STPC Video Controller can be configured to generate video timing (driving HSYNC# and B/T#) or lock to these signals when generated by an external video timing source. The OSD (On Screen Display) signal is not supported.

15.4. VIP SPECIFICATIONS NOT SUPPORTED

15.4.1. ANCILLARY DATA

The Video Input Port does not support the capture of Ancillary Data. The VIP specification allows this method to be used for capturing sliced VBI and digital audio PCM data through the Video Input Port.

Sliced VBI data cannot be transferred as ancillary data. It is intended that sliced VBI data be transferred from VIP compliant devices using the Host Port since only this method is available during MPEG playback.

Digital Audio PCM data cannot be transferred to the Frame Buffer. The specification for audio data transfer as Ancillary Data is still being developed by an ITU task force.

15.4.2. DMA CHANNEL RESTRICTIONS

The DMA controllers are only capable of accessing Host FIFO Space. They are not capable of accessing Host Register Space.

15.4.3. CHROMA MASK

There is no support provided for chroma key mixing since the STPC CRTC supports both colour and chroma based video mixing.

15.5. VIDEO INPUT MODULE ADDRESS SPACE

VIP Target devices are memory mapped into Graphics Register Space in the 4 Mbyte section allocated to memory mapped graphics and video registers. 256 Kbytes of this space are allocated to the Video Input Module. VIP Host Target Address Space occupies 32 Kbytes of this region. Host Port and DMA registers occupy another 32 Kbytes of this region. The Video Input Module Address Map is shown in Table 15-2.

Table 15-2. Video Input Module Address Map

AD[31:28]	[27:24]	[23:20]	[19:18]	[17:15]	Description
0000	GBASE	0110	00	100	Video Input Port Registers - PIXCLK domain

15.6. VIP VIDEO INPUT PORT REGISTERS

The video input port registers are all initialised to 0 at power up. Writes to registers marked reserved are ignored, reads always return 0.

15.6.1. FRAME BUFFER ADDRESS READBACK

The Frame Buffer address register is loaded from Vid_Ad0 or Vid_Ad1. A read path is provided at this address for testing purposes. The value is unsynchronised and should not be read during active video.

FB1_Adr Access = 8400000h Regoffset = 0x00h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv										FBA					
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FBA															
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-22	Rsv	**Reserved.** This Read-Only field is reserved. When read it returns ‘0’s.
Bits 21-0	FBA	**Frame Buffer Address.** Read-only static readback for Frame Buffer address register (for test). Lower address bits 2-0 are reserved and when read return a value of ‘0’.

15.6.2. VIDEO INPUT PORT CONFIGURATION REGISTER

The top byte of Vin_CFG is reserved for enabling and disabling interrupts. Bits 31-28 are used to reset interrupt enables. Individual interrupts are disabled by writing a ‘1’ to the associated Reset IRQ enable field. Writing a zero to the Reset IRQ enable field preserves the existing enable status. Read values for these fields are undefined and should be masked off before comparing.

Bits 27 to 24 are the interrupt enables. Individual interrupts are enabled by writing a ‘1’ to the associated interrupt enable field. Writing a zero preserves the existing value.

Writing a ‘1’ to both the enable and reset enable field at the same time produces undefined results.

Vin_CFG Access = 8400000h Regoffset = 0x04h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rst_BFIEn	Rst_FIEn	Rst_VBIEn	Rst_BOEn	BF_IEn	F_IEn	VB_IEn	BO_IEn.	VCLK		ST	DE	AU	VIF		
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALEN Rsv	BS	FB1		FDC		FCC			DBE		EVC	FB1			IPE
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bit 31	Rst_BFIEn	**Reset Buffer Full IRQ Enable** (Write-only) 0: Preserve Buffer Full IRQ enable 1: Reset Buffer Full IRQ enable
Bit 30	Rst_FIEn	**Reset Field IRQ Enable** (Write-only) 0: Preserve Field IRQ enable 1: Reset Field IRQ enable
Bit 29	Rst_VBIEn	**Reset Vertical Blank IRQ Enable** (Write-only) 0: Preserve Vertical Blank IRQ enable 1: Reset Vertical Blank IRQ enable
Bit 28	Rst_BOEn	**Reset Buffer Overflow IRQ Enable** (Write-only) 0: Preserve Buffer Overflow IRQ enable 1: Reset Buffer Overflow IRQ enable
Bit 27	BF_IEn	**Buffer Full IRQ Enable** 0: Preserve existing BF_IEn value 1: IRQ is generated when either video input buffer goes full
Bit 26	F_IEn	**Field Change IRQ Enable** 0: Preserve existing F_IEn value 1: IRQ is generated when the internal Field bit changes
Bit 25	VB_IEn	**Vertical Blank IRQ Enable** 0: Preserve existing VB_IEn value 1: IRQ is generated at the end of the current field after flushing the frame buffer FIFO

Bit Number	Mnemonic	Description
Bit 24	BO_IEn.	**Video Input Buffer Overflow Enable** 0: Preserve existing BO_IEn value 1: IRQ is generated when either video input buffer overflows (see vin_stat bit 24)
Bits 23-22	VCLK	**VCLK source** (see Table 15-3) VCLK source determines the clock source for the Video Input Port. A clock is required for the Video Input Port to respond to host accesses. The power on default is MCLK. VCLK is only an output when DCLK is the enabled source and the video port clock and timing signals are being generated by the CRTC. The following sequence is recommended when enabling the external VCLK. 1. Set vin_cfg[23:22] to '00' (Select MCLK for the internal timing). This insures that the VCLK pin is not being driven by the Video controller. 2. Enable the external VCLK driver. 3. Set vin_cfg[23:22] to '01' (Select VCLK for internal timing). This resets the time-out counter and selects the VCLK input. If the time-out counter (~16 MCLK periods) expires without detecting a valid VCLK input, the clock source will be changed back to MCLK. 4. Check vin_stat[9] to make sure that VCLK is present. **Note:** If video is not being captured correctly, vin_stat[9] should be checked to be sure that a valid VCLK is being provided.
Bit 21	ST	**Start Buffer, Start_BF.** Controls which video input buffer will be filled first. 0: Video input buffer 0 filled first 1: Video input buffer 1 filled first
Bit 20	DE	**Decimator Enable, Dec_En.** 0: No decimation of input pixels 1: Enable 2:1 video decimator
Bit 19	AU	**Auto Update Auto_Up.** This bit enables automatic updating of the displayed video buffer. 0: Buffer must be updated by the driver. 1: Buffer automatically switched to the most recently completed display buffer.
Bits 18-16	VIF	**Video Input Format, VI_Form.** This field controls how the video stream will be decoded. (See Table 15-4) In VIP 1.0 Video Mode the timing information is recovered from SAV and EAV codes embedded in the video stream. The video timing generator may be used to generate system timing In multiplexed CCIR-601 mode, the video input port can generate video timing or lock to an external source. Video timing generation is enabled for formats 0-3 and disabled for modes 4-7.
Bit 15:	ALEN	**Address Limit Enable**: 0: Disable 1: Address Limitation Enable Enabled to prevent corrupt input video data overwriting memory. Makes use of limit address registers LADDR0 in Section 15.6.4. and LADDR1 in Section 15.6.5.

Bit Number	Mnemonic	Description
Bit 14	BS	**Byte Swap.** 0: Bytes within words are swapped on input to match the format expected by the video display, Y Cb Y Cr (default setting) 1: Bytes within words are passed directly through the input port. This setting should be selected in pass through mode to maintain Cb Y Cr Y format.
Bits 13-12	FB1	**FB1 High Water Mark.** HIGH ORDER 2 BITS of the Frame Buffer FIFO high water mark. Vin_cfg[13-12] and vin_cfg[3-1] are concatenated to form the video Frame Buffer FIFO high water mark. Video data is buffered between the video input port and the Frame Buffer in a FIFO (FB1). FB1 High Water Mark is used to optimise Frame Buffer accesses by specifying the point where FB1 makes a request to the Frame Buffer memory controller. When the Frame Buffer FIFO contains this number of QWORDs it will request access to the Frame Buffer.
Bits 11-10	FDC	**Frame Drop Control.** Frame Drop Control determines how often frames are captured. A Frame period consists of an odd and even field sequence, even when only one of the fields is captured. Frame dropping can be used to reduce the input video stream bandwidth when bottlenecks prevent capture and/or transmission at full video rates. (See Table 15-5)
Bits 9-7	FCC	**Field Capture Control.** Field Capture Control determines what fields are used to generate a frame. In progressive scan mode, fields are de-interlaced by merging odd and even fields into a single video buffer. This method of de-interlacing provides the highest vertical resolution but can cause motion artifacts where there are areas of movement. When capturing interlaced video in double buffer mode, the buffer is switched at the end of each enabled field (see Table 15-6)
Bit 6	DBE	**Double Buffer Enable.** Double Buffer Enable allows the amount of Frame Buffer memory to be reduced when capturing at less than full video rates. When single buffering is selected, all captured fields are written to the Frame Buffer using the buffer selected by the start buffer field 0: Single buffer 1: Double buffer
Bit 4	EVC	**Enable Video Capture.** Enable Video Capture starts or stops video capture operation. Video capture starts at the first enabled field of the next frame when video is being captured based on the field bit. When both fields are enabled, frame capture starts with field 1 (the odd field) as defined by ITU-R 656 . 0: Video capture will end after current frame 1: Video capture will begin at start of next frame or task
Bits 3-1	FB1	**FB1 High Water Mark.** LOW ORDER 3 BITS of the Frame Buffer FIFO high water mark.. See vin_cfg[13:12] for the high order bits.
Bit 0	IPE	**Input Port Enable.** The Input Port Enable allows the port to be reset by software. This bit should not be asserted during normal operation as it unconditionally resets the port to the default values . 0: Video input port disabled, counters/state machines initialised, capture of video stopped. 1: Video input port enabled

Table 15-3. VCLK Source

Bit 23	Bit 22	VCLK source
0	1	Use MCLK for video timing generator and interface clock (default).
0	1	Use input VCLK for video interface clock.

1	0	Use DCLK for video interface clock.
1	1	Reserved

Table 15-4. Video Input Format

Bit 18	Bit 17	Bits 16	Video Input Format
0	0	0	VIP 1.0 Video Mode
0	0	1	Reserved
0	1	0	Pass through
0	1	1	STi3520A Compatibility mode (Multiplexed CCIR-601)
1	x	x	Reserved

Table 15-5. Frame Capture/Drop

Bit 11	Bits 10	Frame Capture/Drop
0	0	Capture all frames
0	1	Capture first frame, drop one, repeat
1	0	Capture first frame, drop two, repeat
1	1	Capture first frame, drop three, repeat

Table 15-6. Field Capture Control

Bit 9	Bit 8	Bits 7	Field Capture Control
0	0	0	Reserved
0	0	1	Interlaced mode, capture only odd fields
0	1	0	Interlaced mode, capture only even fields
0	1	1	Interlaced mode, capture both even and odd fields

15.6.3. VIDEO INPUT PORT STATUS REGISTER

Status bits that are latched are cleared by writing to vin_stat with a bit pattern that contains a ‘1’ in the locations that are being reset and ‘0’ in the locations that are to be preserved. Read only bits are unaffected by write cycles. Reserved bits are undefined and must be masked off before making comparisons.

vin_stat Access = 8400000h Regoffset = 0x08h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv				Ch2Int	Ch1Int	VCLK	VB	CP	F IRQ	VB IRQ	OEF	AB	BO IRQ	BF	B 0 F
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-12	Rsv	**Reserved**. This Read-Only field is reserved. When read it returns undefined data.
Bit 11	Ch2Int	**Channel 2 Interrupt Pending Copy, Ch2Int.** This bit is a read-only copy of the interrupt request bit in the DMA channel status register. It is provided here for convenience. See the description under DMA operation for more information.
Bit 10	Ch1Int	**Channel 1 Interrupt Pending Copy, Ch1Int.** This bit is a read-only copy of the interrupt request bit in the DMA channel status register. It is provided here for convenience. See the description under DMA operation for more information.
Bit 9	VCLK	**VCLK present.** This read-only bit reflects the presence of the VCLK signal 0: VCLK is not present 1: VCLK is present
Bit 8	VB	**Vblank.** This read-only bit reflects the value of the internal vertical blank 0: Active video region 1: Vertical blanking region
Bit 7	CP	**Capture in Progress.** This read-only bit is set at the start of the first video frame after Enable Capture of Video is set. It is cleared at the end of the first frame after Enable Capture of Video is cleared. This bit is controlled by hardware.
Bit 6	F IRQ	**Field IRQ.** This bit is set and latched when the digital field bit changes and the Field IRQ enable bit (vin_cfg) is set to 1. It is cleared by writing a value of 1 to vin_stat[6].
Bit 5	VB IRQ	**VBlank IRQ**. This bit is set and latched when the last line of enabled video has been written to the Frame Buffer and the vblank IRQ enable bit (vin_cfg) is set to ‘1’. It is cleared by writing a value of ‘1’ to vin_stat[5].

Bit Number	Mnemonic	Description
Bit 4	OEF	**Odd/Even Field.** This is a read only bit that reflects the value of the internal field flag (as defined by ITU-R 656). This status bit is not affected by the B/T# inversion control 0: Field 1 (Top field) is currently being processed 1: Field 2 (Bottom field) is currently being processed
Bit 3	AB	**Active Buffer.** This is a read only bit that reflects the value of an internal flag. It indicates which video buffer is currently being filled.
Bit 2	BO IRQ	**Buffer Overrun IRQ.** This bit is set and latched when either of the buffers receives a write from the pixel packer and the corresponding buffer full flag is set, indicating that a buffer overrun has occurred. If the corresponding interrupt enable is asserted, an interrupt is generated. This bit is cleared by writing a value of ‘1’ to vin_stat[2].
Bit 1	BF	**Buffer 1 Full.** This bit is set and latched when buffer 1 receives the last pix-el of a captured frame. This condition can, if enabled, generate an IRQ This bit is cleared by writing a value of ‘1’ to vin_stat[1].
Bit 0	B 0 F	**Buffer 0 Full.** This bit is set and latched when buffer 0 receives the last pix-el of a captured frame. This condition can, if enabled, generate an IRQ. This bit is cleared by writing a value of ‘1’ to vin_stat[0].

15.6.4. VIDEO INPUT BUFFER ADDR 0

Lines of video always start at a quad word boundary in the Frame Buffer. When the display window size is not a multiple of eight, any remaining bytes in the last quad word will be unused (and undefined).
The LS three bits of this register are hardwired to zero to force QWORD alignment.

vin_ad0 Access = 8400000h Regoffset = 0x0Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv										VBA 0					
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VBA 0															
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-22	Rsv	**Reserved**. This Read-Only field is reserved. When read it returns ‘0’s
Bits 21-0	VBA 0	**Video Buffer Addr 0.** Quad word Frame Buffer start address for video input buffer 0. Lower address bits 2-0 are reserved and when read return a value of ‘0’.

15.6.5. VIDEO INPUT BUFFER ADDR 1

The Least Significant three bits of this register are hardwired to zero to force QWORD alignment.

vin_ad1 Access = 8400000h Regoffset = 0x10h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv										VBA 1					
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VBA 1															
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-22	Rsv	**Reserved**. This Read-Only field is reserved. When read it returns ‘0’s
Bits 21-0	VBA 1	**Video Buffer Addr 1.** Quad word Frame Buffer start address for video input buffer 1. Lower address bits 2-0 are reserved and when read return a value of ‘0’.

15.6.6. VIDEO INPUT DEST PITCH

When the Field Capture Control selects one of the interlaced modes, the destination pitch is set to the number of quad words required to hold a line of video data. When de-interlacing by merging odd and even fields is selected, the destination pitch should be set to twice the number of quad words required to hold a line of video data. The Least Significant three bits of this register are hardwired to zero to force QWORD alignment.

vin_dp Access = 8400000h Regoffset = 0x14h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv		DP													
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-14	Rsv	**Reserved**. This Read-Only field is reserved. When read it returns ‘0’s.
Bits 13-0	DP	**Destination Pitch.** This register holds the number of bytes in the Frame Buffer the beginning of one video scan line to the next. Lower address bits 2-0 are reserved and when read return a value of ‘0’.

15.6.7. EXTERNAL TIMING GENERATOR 1

vtg_ext1 Access = 8400000h Regoffset = 0x28h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VGT	Rsv	Bt_oe	HSYNC_OE	Rsv	Bt_pol	HSYNC_POL	Rsv	GM			Rsv			HS_St	
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HS_St				Rsv			HS_End								HS_Odd
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bit 31	VGT	**VTG enable.** 0: Video timing is reset to the start of field 1 1: Video timing generator is enabled
Bit 30	Rsv	**Reserved.** This Read-Only field is reserved. When read it returns undefined data.
Bit 29	Bt_oe	**Output enable for the B/T# video timing signal , bt_oe.** Set to ‘1’ when the Video Controller is generating the system video timing signals. 0: B/T# is an input 1: B/T# is an output
Bit 28	HSYNC_OE	**Output enable for the HSYNC- video timing signal, hsync_oe.** Set to ‘1’ when the Video Controller is generating the system video timing signals. 0: HSYNC# is an input 1: HSYNC# is an output
Bit 27	Rsv	**Reserved.** This Read-Only field is reserved. When read it returns undefined data.
Bit 26	Bt_pol	**B/T# polarity, bt_pol.** This bit defines the active edge for the B/T# signal as input and output. The setting of this bit sets the polarity of the external signal to be high or low true. The polarity of the internal field (F) and horizontal blank (H) bits are not affected. 0: B/T# is low for field 1 1: B/T# is high for field 1
Bit 25	HSYNC_POL	**HSYNC# polarity, hsync_pol.** This bit defines the active edge for the HSYNC# signal as input and output. The setting of this bit sets the polarity of the external signal to be high or low true. The polarity of the internal field (F) and horizontal blank (H) bits are not affected by these bits. 0: HSYNC# is low true 1: HSYNC# is high true
Bit 24	Rsv	**Reserved.** This Read-Only field is reserved. When read it returns undefined data.
Bits 23-21	GM	**Genlock Mode.** Defines the method for genlocking to an external source. (See Table 15-7).

Bit Number	Mnemonic	Description
Bits 20-18	Rsv	**Reserved.** This Read-Only field is reserved. When read it returns undefined data.
Bits 17-12	HS_St	**Leading edge of HSYNC# in pixels, HS_St.** Allows the HSYNC# trailing edge to be shifted relative to the start of the horizontal line in pixels, referenced to the horizontal counter. Since the video clock runs at twice the pixel rate, this value must be multiplied by 2 (shifted left 1) before being compared to the horizontal count. HS_Odd is used to generate the least significant bit, insuring that HSYNC# can be generated at any point during the horizontal scan line.
Bits 11-9	Rsv	**Reserved.** This Read-Only field is reserved. When read it returns undefined data.
Bits 8-1	HS_END	**Trailing edge of HSYNC# in pixels, HS_End.** Allows the HSYNC# trailing edge to be shifted relative to the end of a horizontal line in pixels, referenced to the horizontal counter. Since the video clock runs at twice the pixel rate, this value must be multiplied by 2 (shifted left 1) before being compared to the horizontal count. HS_Odd is used to generate the least significant bit, insuring that HSYNC# can be generated at any point during the horizontal scan line.
Bit 0	HS_ODD	**HSYNC# Odd compensation, HS_Odd.** This bit allows the leading and trailing edges of HSYNC# to be shifted by 1 VCLK to compensate for an odd number of pipe stages between the video input port and an external device. This bit is used as the least significant bit of HS_End and HS_St when being compared to the horizontal counter. When the video timing generator is in master mode, the leading edge of the external HSYNC occurs on the clock edge following when the horizontal counter matches the HSSt value. The trailing edge occurs on the clock edge following when the horizontal counter matches the HSEnd value. When the video timing generator is in slave mode, the horizontal counter is set to HSSt value on the second VCLK edge following HSYNC# assertion. In slave mode, the horizontal timing is independent of the trailing edge of HSYNC# and HSEnd is ignored. The default values are specified to match ITU-R 656 (525 line) timing in slave mode.

Table 15-7. Genlock Mode

Bit 23	Bit 22	Bit 21	Genlock Mode
0	0	0	No genlocking. Video timing generator resets horizontal and vertical counters based on H_Total and V_Total. (default)
0	0	1	Genlock to B/T# and HSYNC#
0	1	0	Genlock to SAV/EAV codes
		1xx	Reserved

15.6.8. EXTERNAL TIMING GENERATOR 2

vtg_ext2 Access = 8400000H Regoffset = 0x2Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv											BT_Dly1				
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BT_Dly1					BT_Dly2										BT_Odd
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-21	Rsv	**Reserved**. This Read-Only field is reserved. When read it returns ‘0’s.
Bits 20-11	BT_Dly1	**B/T# delay for field 1 in pixels, BT_Dly1.** This field determines the location relative to the horizontal counter that B/T# switches to indicate field 2 of an interlaced frame.
Bits 10-1	BT_Dly2	**B/T# delay for field 2 in pixels, BT_Dly2.** This field determines the location relative to the horizontal counter that B/T# switches to indicate field 2 of an interlaced frame.
Bit 0	BT_Odd	**B/T## Odd compensation, BT_Odd.** This bit allows the leading and trailing edges of B/T# to be shifted by 1 VCLK to compensate for an odd number of pipe stages between the video input port and an external device. This bit is used as the least significant bit of BT_Dly1 and BT_Dly2 when being compared to the horizontal counter.

15.6.9. HORIZONTAL TIMING GENERATOR

vtg_ht Access = 8400000H Regoffset = 0x30h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
H_Start										Rsv	H_End				
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
H_End					Rsv	H_Total									
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-22	H_Start	**Horizontal start of active video in pixels, H_Start.** When video capture is based on the video timing generator, The H_Start and H_End values are used to determine when video is captured within the vertical display window.
Bit 21	Rsv	**Reserved.** This Read-Only field is reserved. When read it returns undefined data.
Bits 20-11	H_End	Horizontal end of active video in pixels, H_End.
Bit 10	Rsv	**Reserved.** This Read-Only field is reserved. When read it returns undefined data.
Bits 9-0	H_Total	**Total number of horizontal pixels, H_Total.** This field contains the total number of pixels per line.

15.6.10. VIDEO TIMING GENERATOR

V_Start specifies the first line of active video in each field.

V_End specifies the last line of active video in each field.

vtg_vt Access = 8400000h Regoffset = 0x34h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
V_Start										Rsv	V_ End				
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
V_End					Rsv	V_Total									
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-22	V_Start	**Vertical Field Start, V_ Start.** Line number of the last line of blanked video in each field. The first line of active video is V_Start + 1
Bit 21	Rsv	**Reserved.** This Read-Only field is reserved. When read it returns undefined data.
Bits 20-11	V_End	Vertical Field End, V_ End. Line number of the last line of active video in each field.
Bit 10	Rsv	**Reserved.** This Read-Only field is reserved. When read it returns undefined data.
Bits 9-0	V_Total	**Vertical Total, V_Total.** V_Total contains the total number of lines in a field. When the internal vertical counter reaches the value contained in V_Total it restarts from either count zero or count one, depending on the field. It gets reset to 1 at the beginning of field 1, making the number of lines in field 1 equal to V_Total. At the beginning of field 2 it gets reset to zero making the number of lines V_Total + 1. The internal field bit (F) gets inverted coincident with the resetting of the vertical and horizontal counters.

15.6.11. LIMIT ADDRESS REGISTERS

Fill the limit address registers of frame buffers 0 and 1. For details on how to activate this feature, see Section 15.6.2. "Video Input Port Configuration Register"

Laddr0 Access = 8400000h Regoffset = 0x3Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv													Laddr0[21:3]		
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Laddr0[21:3															
Default value after reset = 00000000h															

Laddr1 Access = 8400000h Regoffset = 0x38h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv													Laddr1[21:3]		
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Laddr1[21:3															
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31:19	Rsv	**Reserved.** This Read-Only field is reserved. When read it returns undefined data.
Bits 18:0	Laddr0/1	**Limit Address Registers of Frame Buffers 0 and 1:** When the current address is greater than the limit address, this address is then held at the same value until the end of capture of the given field is reached, i.e. until vertical blanking occurs. This function, enabled by bit 15 of the Video Input Port Configuration register, is used to prevent memory overrun. When not enabled, corrupted video input data can cause the VMI module to continue to write to memory by incrementing the address, thereby causing memory to be over-written.

16. VIDEO PIPELINE REGISTERS

16.1. INTRODUCTION

The STPC Atlas video input signals, buffering and video output signals are managed by the Video Controller, where the video input and display signals are controlled through the Video Pipeline registers. These registers, which control the settings for the display buffer areas, plus filter control, colour mixing and colour space mixing, are described in the remainder of this Chapter.

The Video Input signal and buffering are controlled by the Video Input Port, which is described in the separate Video Input Port Chapter.

16.2. VIDEO PIPELINE REGISTER LOCATIONS

The Video Pipeline registers, similar to the extended graphics (non-VGA) registers, are located in the 4-MByte memory-mapped registers region of the 16-MByte memory space occupied by the Graphics Controller. The Video Pipeline registers are located at the 256-KByte wide sub-region 2. The figure below shows the address format for the Video Pipeline registers.

4Mbyte Region 1

256KbyteSub-region2

31 27	26 24	23 22	21 18	17 16
00001	<GBASE>	01	0010	01

All registers can be read with accesses of any width. The CPU can read any register via byte (8-bit), word (16-bit), or double-word (32-bit) accesses. Writes must be done using double-word (32-bit) transfers.

16.3. .SOURCE SPECIFICATION REGISTERS

16.3.1. VIDEO SOURCE BASE REGISTER

This register specifies the DRAM linear starting address of the source image, aligned to an eight byte boundary. This address may specify either the top left corner or bottom left corner, depending on the state of the Y_Vid_Src_dir bit in the Video_Src_Pitch register.

This register is double buffered, the active register is only updated during vertical blanking.

Video_Src_Base Access = X480000h Regoffset = 000h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv										VSI					
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VSI															
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-22	**Rsv**	**Reserved**
Bits 21-0	**VSI**	**Base linear address of the Video Source Image** Lower address bits 2-0 are reserved and when read return a value of ‘0’.

16.3.2. VIDEO SOURCE PITCH REGISTER

This register contains the Video_Src_pitch field, which specifies the number of bytes which must be added to the address of a pixel on one line of the video source image to compute the address of the corresponding pixel on the line below.

This register also contains the Y_Vid_Src_dir bit which specifies the Y direction in which the Video Source Image should be read, and the Video_Colour_fmt field which defines the colour format of the Video Source Image.

Video_Src_Pitch Access = X480000h Regoffset = 0x04h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv		VSI		Y_V_S_D	Current_Ad										
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-14	**Rsv**	**Reserved**
Bits 13-12	**VSI**	**Video Source Image colour format:** 00 - RGB 555 01 - RGB 565 10 - YUV 422

Bit 11	**Y_V_S_D**	**Y_Vid_Src_dir** Specifies the Y direction in which the Video Source Image should be read. This bit controls the translation of XY addresses to linear DRAM addresses.
		If(Y_Vid_Src_dir == 0)
		DRAM linear address = Video_Src_Base
		+ (YDIFF * Vid_Src_Pitch);
		Else
		DRAM linear address = Video_Src_Base
		- YDIFF * Vid_Src_Pitch);
		Where YDIFF is a 1 bit value that varies.

Bits 10-0	**Current_Ad**	Specifies the amount to add to the current address to get to the address of the corresponding pixel in the next line. Lower address bits 2-0 are reserved and when read return a value of ‘0’.

16.3.3. VIDEO SOURCE DIMENSION REGISTER

This register contains the dimensions of the Video Source Image relative to the starting corner.

This register is double buffered. The active register is only updated during vertical sync.

Vid_Src_dim Access = X480000h Regoffset = 0x08h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv						dY									
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv						dX									
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-26	**Rsv**	**Reserved**
Bits 25-16	**dY**	**dY**, the height of the Video Source image - 1, in lines, from the starting corner to the end of the image (dependent on Y_Vid_Src_dir)
Bits 15-10	**Rsv**	**Reserved**
Bits 9-0	**dX**	**dX,** the width in pixels, of a line

16.3.4. CRTC BURST LENGTH REGISTER

This register contains the CRTC low water mark and burst length.

CRTC_Burst_length Access = X480000h Regoffset = 0x0Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Crtc_Dlwm								Crtc_lwm							
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Crtc_dt					Rsv	Crtc_Bl									
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-24	**Crtc_Dlwm**	**Delta low water mark, crtc_dlwm** Together with crtc_dt and crtc_lwm, this field defines a variable low water mark. When the video window starts, the CRTC low water mark is set to crtc_lwm. After that time, for every crtc_dt*8 pixels' time elapsed, the low water mark will be incremented by crtc_dlwm bytes. Since this field is represented as a 2's complement number, setting bit 31 results in a low water mark which is a decreasing function of time. A decreasing or constant function will be the normal mode of operation of the CRTC low water mark during the video window. Note that this CRTC low water mark is distinct from the one described in CR1B. This one is valid during the video windows only. For normal CRTC operation (scanlines or pixels outside the video window), the pertinent CRTC low water mark is specified by CR1B. Guarantee of the CRTC ownership can be achieved by the Setting of this field to zero. This causes the CRTC low water mark to remain at a constant value of crtc_lwm.
Bits 23-16	**Crtc_lwm**	**crtc_lwm,** the (initial) low water mark for the CRTC FIFO in bytes. During the video window, if the CRTC FIFO occupancy rises above the low water mark (defined as a function of time by crtc_dlwm and crtc_dt) and the video occupancy rises above the video low water mark then ownership of the system DRAM can be given back to the CPU. The value of this register field can only be a multiple of eight (bits 18-16 are not writable and read as zeroes).
Bits 15-11	**Crtc_dt**	**crtc_dt, delta t** See the description of crtc_dlwm above.
Bit 10	Rsv	*Reserved*.
Bits 9-0	**Crtc_Bl**	**Minimum CRTC burst length, crtc_bl** This is the minimum number of bytes that will be sent in one transfer to fill the CRTC FIFO (during the active video window only). This value can only be a multiple of eight (bits 2-0 are not writable and read as zeroes) since pixels are fetched 8 bytes at a time.

16.3.5. VIDEO BURST LENGTH REGISTER

This register is the video counterpart of the previous register. It specifies the video low water mark and burst length.

Vid_Burst_length Access = X480000h Regoffset = 0x10h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Vid_Dlwm								Vid_lwm							
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Vid_Dt					Rsv	Vid_Bl									
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-24	**Vid_Dlwm**	**Delta low water mark, vid_dlwm** Together with vid_dt and vid_lwm, this field defines a variable low water mark. When the video window starts, the video low water mark is set to vid_lwm. After that time, for every vid_dt*8 pixels' time elapsed, the low water mark will be incremented by vid_dlwm bytes. As with crtc_dlwm, above, This field is a 2's complement number. Setting this field to zero causes the video low water mark to remain at a constant value of vid_lwm.
Bits 23-16	**Vid_lwm**	**vid_lwm,** the (initial) low water mark for the video FIFO in bytes. During the video window, if the video FIFO occupancy rises above the low water mark (defined as a function of time by vid_dlwm and vid_dt) and the crtc occupancy rises above crtc_lwm then ownership of the system DRAM can be given back to the CPU. The value of this register field can only be a multiple of eight (bits 18-16 are not writable and read as zeroes).
Bits 15-11	**Vid_Dt**	**vid_dt, delta t** See the description of vid_dlwm above.
Bit 10	**Rsv**	*Reserved*
Bits 9-0	**Vid_Bl**	**Minimum video burst length, vid_bl** This is the minimum number of bytes that will be sent in one transfer to fill the video FIFO. This value can only be a multiple of eight (bits 2-0 are not writable and read as zeroes) since pixels are fetched 8 bytes at a time.

16.3.6. DESTINATION SPECIFICATION REGISTERS

16.3.6.1. Video Destination XY Register

This register contains the coordinates of the top left corner of the video window.

This register is double buffered. The active register is only updated during vertical sync.

Vid_Dst_XY Access = X480000h Regoffset = 0x14h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv						Y									
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv					X										
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-26	**Rsv**	**Reserved**
Bits 25-16	**Y**	**Y,** the Y coordinate of the top edge of the video window, relative to the display. The first display line is line 0.
Bits 15-11	**Rsv**	**Reserved**
Bits 10-0	**X**	**X,** the X coordinate of the left edge of the video window, relative to the display. The first pixel of each display line is pixel 0.

16.3.6.2. Video Destination Dimension Register

This register contains the dimensions of the video window.

This register is double buffered. The active register is only updated during vertical sync.

Vid_Dst_dim Access = X480000h Regoffset = 0x18h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv						dY									
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv					dX										
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-26	**Rsv**	**Reserved**
Bits 25-16	**dY**	**dY,** the height of the video window in screen lines - 1 is entered in this field.
Bits 15-11	**Rsv**	**Reserved**
Bits 10-0	**dX**	**dX,** the width of the video window in screen pixels.

16.4. FILTER CONTROL REGISTERS

16.4.1. HORIZONTAL SCALE REGISTER

This register contains the control for horizontal scaling and decimation.

This register is double buffered. The active register is only updated during vertical sync.

Horiz_Scale Access = X480000h Regoffset = 0x20h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv											F0E	F1E	Rsv		
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv			HPI												
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-21	**Rsv**	**Reserved**
Bit 20	**F0E**	**Filter Enable 0, F0E.** When in YUV modes, this bit enables the interpolator for the Y channel, and in RGB modes enables all three interpolators. If this bit is 0, pixel replication is used.
Bit 19	**F1E**	**Filter Enable 1, F1E.** When in YUV modes, this bit enables the interpola-tors for the U and V channels. If this bit is 0, pixel replication is used. This bit controls the RGB components in RGB modes and should be the same as F0E.
Bits 28-16	**Rsv**	**Reserved**
Bits 15-13	**Rsv**	**Reserved**
Bits 12-0	**HPI**	**Horizontal Phase Increment, hpi.** Defines the horizontal scale factor. hpi is calculated from source width and destination width:- hpi = (source_width * 4096)/dest_width Note that the maximum value hpi is 4096. Downscaling is not supported

16.4.2. VERTICAL SCALE REGISTER

This register contains the control for vertical scaling and decimation.

This register is double buffered. The active register is only updated during vertical sync.

Vert_Scale Access = X480000h Regoffset = 0x28h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv											VF0E	VF1E	Rsv		
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv			VPI												
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-21	**Rsv**	**Reserved**
Bit 20	**VFOE**	**Vertical Filter Enable 0, VF0E** When in YUV modes, this bit enables the interpolator for the Y channel, and in RGB modes enables all three interpolators. If this bit is 0, pixel replication is used.
Bit 19	**VF1E**	**Vertical Filter Enable 1, VF1E** When in YUV modes, this bit enables the interpolators for the U and V channels. If this bit is 0, pixel replication is used. This bit controls the RGB components in RGB modes and should be the same as VF0E.
Bits 18-16	**Rsv**	**Reserved**
Bits 15-13	**Rsv**	**Reserved**
Bits 12-0	**VPI**	**Vertical Phase Increment, vpi** Defines the vertical scale factor. vpi is calculated from source height and destination height:- vpi = (source_height * 4096)/dest_height Note that the maximum value for vpi is 4096 Downscaling is not supported

16.4.3. COLOUR SPACE CONVERTER SPECIFICATION REGISTER

This register contains the control for the colour Space Converter.

colour space converter specification Access = X480000h Regoffset = 0x2Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv															CSCE
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-1	**Rsv**	**Reserved**
Bit 0	**CSCE**	**colour Space Converter Enable** When set, YUV data is converted to RGB according to the formula:- R = 1.164(Y - 16) + 1.596(V - 128) G = 1.164(Y - 16) - 0.813(V - 128) - 0.392(U - 128) B = 1.164(Y - 16) + 2.017(U - 128) When clear, pixels are passed through unchanged.

16.5. VIDEO AND GRAPHICS MIXING CONTROL REGISTERS

16.5.1. MIX MODE REGISTER

This register contains the Mix_Mode field which defines the method used to mix graphics and video.

Mix_Mode Access = X480000h Regoffset = 0x30h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv														MM	
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-2	**Rsv**	**Reserved**
Bits 1-0	**MM**	**Mix_Mode,** controls the way in which graphics and video are mixed. **Bit 1 Bit 0** 0 0 Video Window only. The video always appears in a rectangular window which is defined by the Destination Specification registers. 0 1 Video Window with colour Key. The Destination specification is further qualified by the colour Key register. Within the specified video window, if the graphics pixel (pre colour palette) is equal to the value specified by the colour Key register, then the corresponding video pixel is displayed, otherwise the graphics pixel is displayed. Note that in 8-bit graphics modes, only colour_Key[7:0] are used in the comparison and in 16-bit graphics modes, Colour_Key[15:0] are used. 1 0 Video Window with Chroma Key. The destination specification is qualified by the Chroma key registers. Chroma key compares each of the pixel compo nents to independent 'high' and 'low' values (be tween limits compare). If all the selected components are between their limits, then the corresponding graphics pixel is displayed, otherwise the video pixel is displayed. Note that the video pixel can be compared either before or after the colour Space Converter. Note also that the chrom key can be programmed to ignore any or all component values. 1 1 *Reserved*

16.5.2. COLOUR KEY REGISTER

This register contains the colour key value used in colour keying mixing.

CCLR_Key Access = X480000h Regoffset = 0x34h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv								CK							
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CK															
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-24	**Rsv**	**Reserved**
Bits 23-0	**CK**	**Colour_Key,** this value is compared to the graphics pixel to determine whether to display the video pixel in colour key mode. When the graphics is operating in 8-bit per pixel mode, Colour_key[7:0] is compared, when the graphics is operating in 16-bits per pixel, Colour_Key[15:0] is compared and when the graphics is operating in 24-bits per pixel, Colour_Key[23:0] is compared.

16.5.3. CHROMA KEY LOW REGISTER

This register contains the chroma key low limits, the component ignore bits and the colour mode bit used in chroma keying mixing.

CKL Access = X480000h Regoffset = 0x38h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv				CK	IC2	IC1	IC0	CH2L							
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1L								CH0L							
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-28	**Rsv**	**Reserved**
Bit 27	**CK**	**Chroma key mode** 0: Components examined at input to colour space converter (YUV mode) 1: Components examined at output of colour space converter (RGB mode)
Bit 26	**IC2**	**Ignore component 2** If set, component 2 (V or B) is ignored in the chroma key comparison.
Bit 25	**IC1**	**Ignore component 1** If set, component 1 (U or G) is ignored in the chroma key comparison.
Bit 24	**IC0**	**Ignore component 0** If set, component 0 (Y or R) is ignored in the chroma key comparison.
Bits 23-16	**CH2L**	**ch2low,** the low limit against which component 2 is compared during chroma key operations.
Bits 15-8	**CH1L**	**ch1low,** the low limit against which component 1 is compared during chroma key operations.
Bits 7-0	**CH0L**	**ch0low,** the low limit against which component 0 is compared during chroma key operations.

16.5.4. CHROMA KEY HIGH REGISTER

This register contains the chroma key high limits used in chroma keying mixing.

CKH Access = X480000h Regoffset = 0x3Ch

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv								CH2H							
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1H								CH0H							
Default value after reset = 00000000h															

Bit Number	**Mnemonic**	**Description**
Bits 31-24	**Rsv**	**Reserved**
Bits 23-16¬	**CH2H**	**ch2high,** the high limit against which component 2 is compared during chroma key operations.
Bits 15-8	**CH1H**	**ch1high,** the high limit against which component 1 is compared during chroma key operations.
Bits 7-0	**CH0H**	**ch0high,** the high limit against which component 0 is compared during chroma key operations.

The operation of the chroma key can be summarised as follows:-

Let Cn represent component n, n = 0..2

Let Chnlow represent Chlow for component n, n = 0..2

Let Chnhigh represent Chigh for component n, n = 0..2

Kn be the result of the compare for component n, n = 0..2

```
for(n = 0; n < 3; n++)
if((Cn >= Chnlow) && (Cn <= chmhigh))
    Kn = 1
  else
    Kn = 0
If((I0||K0) && (I1||K1) && (I2||K2))
  display graphics pixel
else
  display video pixel
```

16.5.5. STATUS REGISTER

This register contains the enable bit for the video scaler.

Filter_Stat Access = X480000h Regoffset = 0x40h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
V_En	Rsv														
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv															
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bit 31	**V_En**	**vid_enable** Setting the enable bit turns on the video scaler.
Bits 30-0	**Rsv**	**Reserved**

17. TFT INTERFACE

17.1. INTRODUCTION

The TFT (Thin Film Transistor) interface converts signals from the CRT controller into control signals for a TFT Flat Panel. It extends the capability of the VGA controller, allowing generation of the display onto an active matrix LCD Flat Panel. The standards supported are 640 x 480, 800 x 600 and 1024 x 768 active matrix TFT flat panels, with 9-bit, 12-bit or 18-bit colour in single pixel mode and 9-bit colour in dual pixel mode. The TFT interface supports a programmable panel size of up to 1024 by 1024 pixels with programmable image positioning and sizing.

The TFT Interface also supports an external PanelLink^{TM} high speed serial transmitter for interfacing with a high resolution panel.

17.2. FUNCTIONAL DESCRIPTION

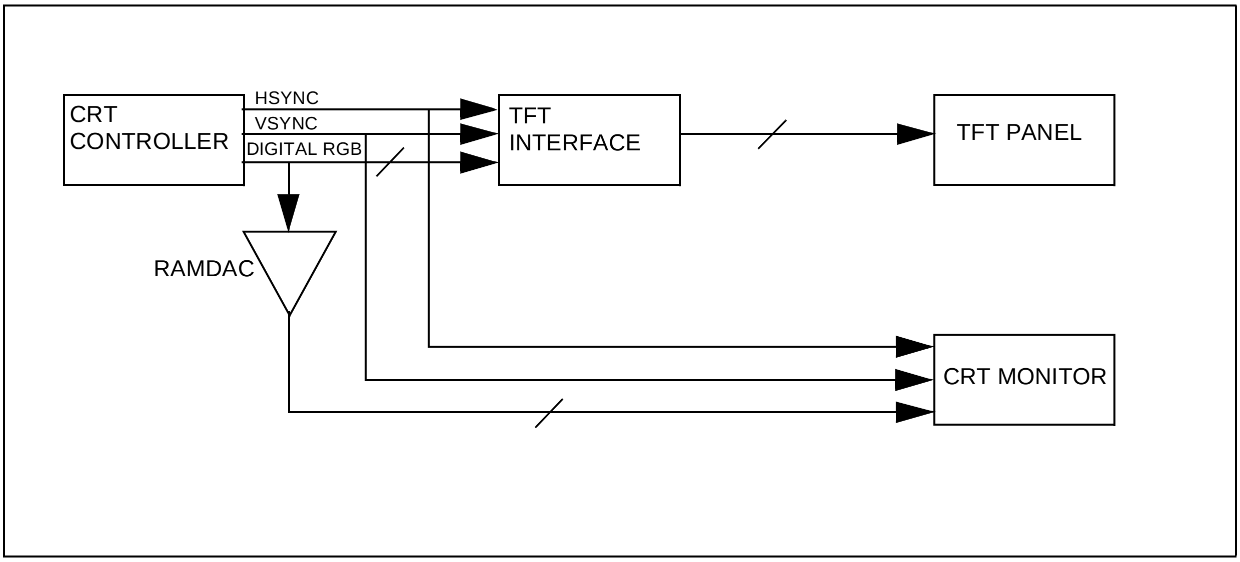
The TFT interface receives the basic timing information via the HSYNC and VSYNC signals from the CRTC interface. It also receives the DOT clock signal from the CRTC, to which all internal TFT interface timings are referenced. The video data from the CRTC data output is also internally taken by the TFT Interface and is stored in a local FIFO buffer. The Flat Panel driver then generates the horizontal and vertical timing signals, the data enable signal, data clock signals and the RGB data applied to the Flat Panel.

The built-in power control block provides a Pulse Width Modulated (PWM) signal for brightness and contrast control and also for the switching of the panel power supply.

17.2.1. TFT INTERFACE DESCRIPTION

Figure 17-1. below shows the TFT interface connected to a CRTC and a TFT panel.

Figure 17-1. TFT Interface



17.2.2. PROGRAMMABLE PANEL SIZE

The height and width of the flat panel is defined in terms of the number of active pixels per scan line and the number of active scan lines per frame respectively. These parameters are programmable through the module configuration registers. The maximum supported panel size is 1024 by 1024.

17.2.3. PIXELS PER CLOCK

TFT panels accept digital RGB input data. For example, a TFT with 9 bits per pixel uses 3 bits to represent red, 3 bits for green and 3 bits for blue. There are two major standards in use for the interface of data to a TFT flat panel: (1) single pixel/clock, where for every DOT, clock data for a single pixel is sent to the panel and (2) two pixels/clock, where data for two pixels are sent to the panel. The advantage of two pixels per clock is that it requires a slower pixel clock, but at the cost of a larger data bus width. The TFT interface module supports 9, 12 and 18-bit interface in single pixel per clock mode and 2 x 9-bit interface for two pixels per clock mode.

Table 17-1. shows the colours displayed by the Screen in single and two pixel per clock mode.

Table 17-1. Pixel Colour Depth

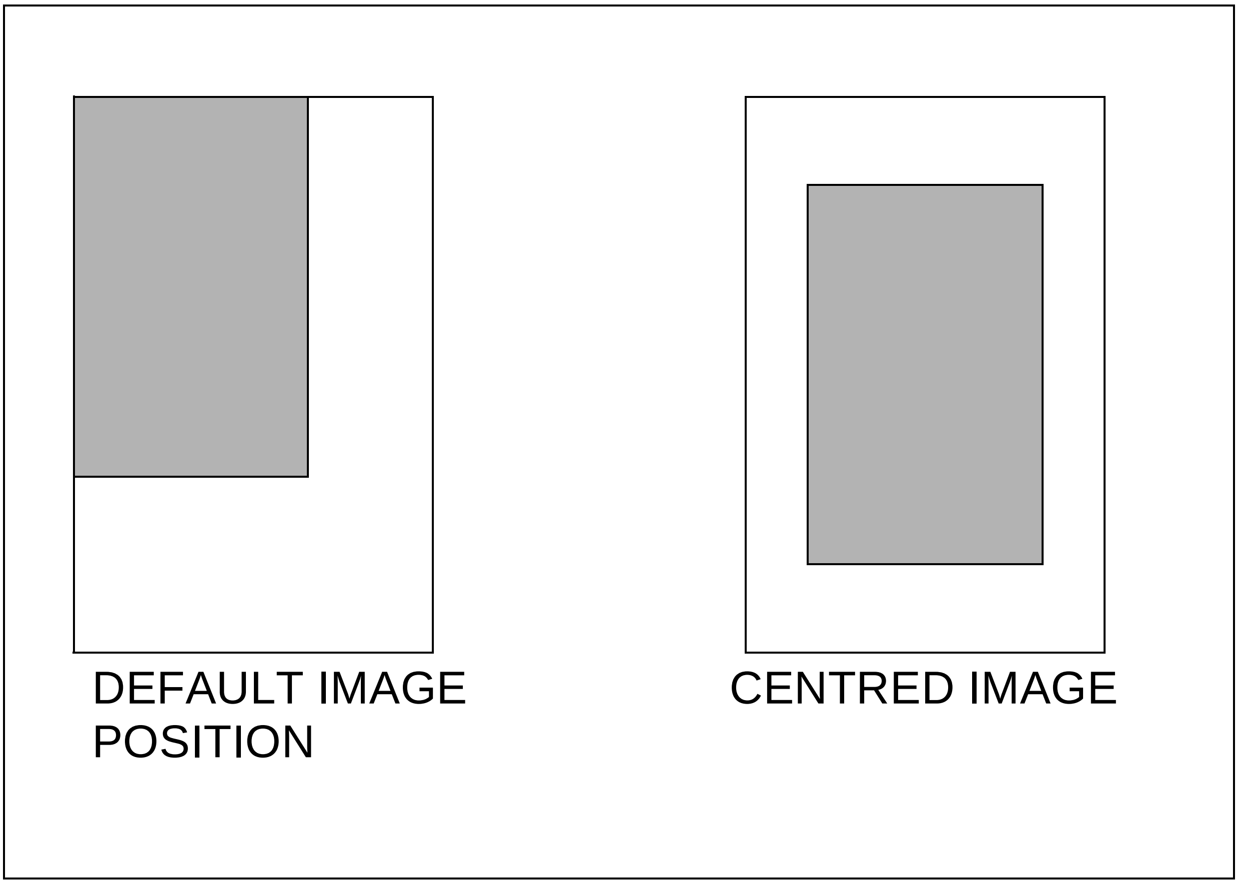
1 Pixel/Clock	2 Pixel/Clock	Signal Type
R[5]	R[2]	odd
G[5]	G[2]	odd
B[5]	B[2]	odd
R[4]	R[2]	even
G[4]	G[2]	even
B[4]	B[2]	even
R[3]	R[1]	odd
G[3]	G[1]	odd
B[3]	B[1]	odd
R[2]	R[1]	even
G[2]	G[1]	even
B[2]	B[1]	even
R[1]	R[0]	odd
G[1]	G[0]	odd
B[1]	B[0]	odd
R[0]	R[0]	even
G[0]	G[0]	even
B[0]	B[0]	even

17.2.4. PROGRAMMABLE IMAGE POSITIONING

Modern multi-sync CRT monitors stretch automatically images of different resolutions in order to fill up the entire screen with the image. Unlike CRT monitors however, TFT panels have a fixed resolution. Thus when a lower resolution image (say VGA 640 by 480 graphics) is displayed on a higher resolution TFT (say 800 by 600), the image will cover only part of the screen. By default the image is positioned at the top left corner of the screen, which is not attractive from an aesthetic stand point. One way to improve the image appearance is to position the image at the centre of the screen (see Figure 17-2. below).

The TFT interface has configuration registers to position the image anywhere on the screen. Changing image position also requires reprogramming of some of the CRT Controller timing registers. For compatibility reasons, it is preferable to use the VGA Controller CRTC which supports shadowing of its timing control registers.

Figure 17-2. Image centring



17.2.5. PROGRAMMABLE BLANK-SPACE INSERTION IN TEXT MODES

As described above, by default lower resolution images will cover only a part of a large TFT panel. One way to fill up the entire panel (in text mode) is to introduce blank space between characters. To expand the image vertically, blank lines can be introduced between two character lines; this can be done easily by programming the CRTC character height control register. To expand the image horizontally, the TFT interface module can be programmed to introduce an arbitrary number of blank pixels every 8 or 9 (or whatever is the character width) input pixels.

17.2.6. IMAGE EXPANSION IN GRAPHICS MODE

Another approach to fill up a larger TFT panel display area is the use of image expansion by pixel replication. This approach works better in the graphics mode than it does in the text mode.

In this method the image is expanded vertically by repeating the last scan line every N lines. For example, a 480 line image can be expanded to a 576 line image by repeating a scan line every 5 scan lines.

A similar approach is used to expand the image horizontally. The interface module can be programmed to repeat every Nth pixel M times. For example, with N=5 and M=1, a 640 pixel-wide image can expand to a 768 pixel-wide image. By introducing such vertical and horizontal expansion, almost the entire screen can be filled up while displaying a 640 by 480 image on a 800 by 600 TFT panel.

17.2.7. BRIGHTNESS CONTROL USING PWM

A programmable Pulse Width Modulated (PWM) signal can be used to directly control the brightness and the contrast of the flat panel without the need of external components. The total pulse width (period) and the duty cycle of this signal are individually programmable through the configuration registers.

17.2.8. PANELLINK TM

PanelLink TM is a proprietary interconnect protocol defined by Silicon Image, Inc. It consists of a transmitter (Sil100) that takes parallel video/graphics data from the host LCD graphics controller and transmits it serially at high speed to the receiver (SIL1101) which controls the TFT panel. The interface uses an adjustable voltage, high speed, serial, DC-balanced, transition minimized, differential transmission technique which reduces the line count, increases physical separation between the panel and the host controller and most importantly lowers EMI.

The TFT interface is designed to support connection of its control signals to the PanelLink transmitter.

17.2.9. FLAT PANEL INTERFACE SIGNALS

TFTDCLK: *DOT Clock Output.*

TFTFRAME: *Vertical Sync. pulse Output.*

TFTLINE: *Horizontal Sync. Pulse Output.*

TFTDE: *Data Enable.*

TFTR5-0: *Red Output.*

TFTG5-0: *Green Output.*

TFTB5-0: *Blue Output.*

TFTENAVDD: *Enable VDD of Flat Panel.*

TFTENVCC: *Enable VCC of Flat Panel.*

TFTPWM: *PWM Back-Light Control.*

17.3. CONFIGURATION REGISTERS

The seven 32-bit configuration registers are listed in Table 17-2. Further details are given in the following paragraphs.

Table 17-2. TFT Interface Configuration Registers

Address	Name	Bits	Description
0x084CC000	Active Input Pixel Count	11:0	Number of pixels in active part of the input scan line
	Input Back Porch	27:16	Input scan line back porch delay
0x084CC004	FP Horizontal Sync Width	11:0	Flat panel horizontal synchronizer active pulse width
	FP Horizontal Back Porch	27:16	Flat panel horizontal back porch
0x084CC008	FP Active Pixel Count	11:0	Number of pixels in active part of the output scan line
	FP Horizontal Front Porch	27:16	Minimum flat panel horizontal front porch
0x084CC00C	FP Vertical Sync Width	11:0	Flat panel vertical synchronizer active pulse width
	FP Vertical Back Porch	27:16	Flat panel vertical back porch
0x084CC010	FP Active Line Count	11:0	Number of active scan lines
	Interface control	27:16	Interface control register
0x084CC014	Blank red	7:0	Red component of a blank pixel
	Blank green	15:8	Green component of a blank pixel
	Blank blue	23:16	Blue component of a blank pixel
	Power control	26:24	Power control for the Flat Panel
	Polarity control	31:27	Polarity control signal
0x084CC018	PWM control	11:0	Pulse width modulation control

17.4. TFT CONFIGURATION REGISTER DESCRIPTIONS

17.4.1. INPUT ACTIVE PIXEL COUNT & BACK PORCH REGISTER

hsync2Dsp1En Access = 0x084CC000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv				Input Horizontal Back Porch											
Default value after reset =															

pxlPerInline

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv				Input Scan Line Active Pixel Count											
Default value after reset =															

Bit Number	Mnemonic	Description
Bits 31-28	**Rsv**	Reserved
Bits 27-16		**Input Horizontal Back Porch.** Width (in units of input DOT clock period) of the input scan line horizontal back porch from the CRTC. Horizontal back porch is the period between horizontal sync going inactive and start of the active display part of the next scan line. A value of zero represents a back porch 2048 DOT clock wide.
Bits 15-12	**Rsv**	Reserved
Bits 11-0		**Active Pixel Count.** Number of pixels in the active part of an input scan line from the CRTC. A value of zero represents 2048 pixels in the active part of input scan line.

17.4.2. FLAT PANEL HORIZONTAL SYNC WIDTH & BACK PORCH REGISTER

hrzntlBckPrch Access = 0x084CC004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv				FP Horizontal Back Porch											
Default value after reset =															

fplinePlsWdth

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv				FP Horizontal Sync Width											
Default value after reset =															

Bit Number	Mnemonic	Description
Bits 31-28	**Rsv**	Reserved
Bits 27-16		**Horizontal Back Porch Width.** Width (in units of input DOT clock period) of the flat panel scan line horizontal back porch of minus three. Therefore a value of one in this register would actually mean a back porch width of four clocks. Horizontal back porch is the period between horizontal sync going inactive and start of DE (Display Enable) of the next scan line. A value of zero represents a back porch 2051 DOT clock wide.
Bits 15-12	**Rsv**	Reserved
Bits 11-0		**FPLINE Pulse Width.** Width (in units of input DOT clock period) of the horizontal sync (FPLINE) pulse generated for the TFT panel. A value of zero makes the pulse width 2048 DOT clock wide.

17.4.3. FLAT PANEL HORIZONTAL ACTIVE PIXEL COUNT & FRONT PORCH REGISTER

hrzntlFrntPrch Access = 0x084CC008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv				FP Horizontal Front Porch											
Default value after reset =															

pxlPerOutLine

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv				FP Horizontal Active Pixel Count											
Default value after reset =															

Bit Number	Mnemonic	Description
Bits 31-28	**Rsv**	Reserved
Bits 27-16		**Minimum Horizontal Front Porch.** The minimum width (in units of input DOT clock period) of the horizontal front porch of flat panel scan line. The actual front porch delay may be greater than this value. Normally the total horizontal width of flat panel scan line will be the same as that of the input scan line. So the horizontal front porch will be total input horizontal width minus the sum of FPLINE width, FP back porch and FP active pixel count. When the vertical scaling through line replication mode is enabled, the time of the active scan lines front porch width will be determined by this register value. Horizontal back porch is the period between DE (Display Enable) going inactive and start of horizontal sync pulse. A value of zero represents a back porch 2048 DOT clock wide.
Bits 15-12	**Rsv**	Reserved
Bits 11-0		**Horizontal Active Pixel Count.** The number of pixels in active part of flat panel scan line. Horizontal back porch is the period between horizontal sync going inactive and start of the active display part of the next scan line. A value of zero represents a back porch of 2048 pixels.

17.4.4. FLAT PANEL VERTICAL SYNC WIDTH & BACK PORCH REGISTER

vrtclBckPrch Access = 0x084CC00C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv				FP Vertical Back Porch											
Default value after reset =															

framePlsWdth

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv				FP Vertical Sync Width											
Default value after reset =															

Bit Number	Mnemonic	Description
Bits 31-28	**Rsv**	Reserved
Bits 27-16		**Vertical Back Porch Width.** The vertical back porch width (in units of horizontal scan line input period) for flat panel frames. A Zero value represents a width of 2048 lines. The vertical back porch is the period between FP-FRAME going inactive and start of the first active scan line of the next frame.
Bits 15-12	**Rsv**	Reserved
Bits 11-0		**FPFRAME Pulse Width.** The active pulse width (in units of horizontal scan line input period) of the vertical sync signal (FPFRAME). A Zero value represents a width of 2048 lines.

17.4.5. FLAT PANEL ACTIVE LINE COUNT & INTERFACE CONTROL REGISTER

Interface Control Access = 0x084CC010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv				EV	DP	IB	RP	Stretch Pixel Period				Real Pixel Period			
Default value after reset =															

totalActvLines

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv				Flat Panel Active Line Count											
Default value after reset =															

Bit Number	Mnemonic	Description
Bits 31-28	**Rsv**	Reserved
Bit 27	**EV**	**Vertical Scaling.** Setting this bit to '1' enables vertical scaling. Setting it to zero disables this feature.
Bit 26	**DP**	**Pixel Clock Data Format.** Setting this bit to '1' indicates two pixel/clock data format, otherwise the data format is assumed single pixel/clock.
Bit 25	**IB**	**Blank Pixel Insert.** If this bit is set to ‘1’ then during the active part of the output scan line, after every N pixels (indicated by real pixel period value), M blank pixels (indicated by stretch pixel period value) will be inserted. Setting it to ‘0’ disables this feature.
Bit 24	**RP**	**Pixel Repeat.** If this bit is set to ‘1’ then during the active part of the output scan line, every Nth pixel (indicated by real pixel period value), will be replicated M times (indicated by stretch pixel period value). Setting it to ‘0’ disables this feature.
Bits 23-20		**Stretch Pixel Period.**
Bits 19-16		**Real Pixel Period.**
Bits 15-12	**Rsv**	Reserved
Bits 11-0		**Active Line Count.** The number of scan lines in the active part of flat panel frames. A Zero value represents a width of 2048 lines.

17.4.6. R,G, B PIXEL, POWER & POLARITY CONTROL REGISTER

Access = 0x084CC014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Polarity Control					Power Control			Blank Pixel Blue Component							
Default value after reset =															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Blank Pixel Green Component								Blank Pixel Red Component							
Default value after reset =															

Bit Number	Mnemonic	Description
Bit 31	**TDP**	When 1 indicates TFT DE polarity is active high. When 0 indicates active low.
Bit 30	**TLP**	**Line Polarity.** Polarity of the flat panel horizontal sync signal (FPLINE). Setting this bit to '1' generates an active high.
Bit 29	**TFP**	**Frame Polarity.** Polarity of the flat panel vertical sync signal (FPFRAME). Setting this bit to '1' generates an active high.
Bit 28	**TVP**	**Vertical Synch Polarity.** Polarity of the vertical sync input signal (VSYNC) from the CRTC. Setting this bit to ‘1’ indicates an active high VSYNC signal.
Bit 27	**THP**	**Horizontal Synch Polarity.** Polarity of the horizontal sync input signal (HSYNC) from the CRTC. Setting this bit to '1' indicates an active high HSYNC signal.
Bit 26	**Evee**	**EVCC.** When set '1', this bit enables the VCC supply to the flat panel. The VCC supply is used for the digital part of the TFT.
Bit 25	**Evdd**	**EVDD.** When set '1', this bit enables the VDD supply to the flat panel. The VDD supply is used for the analog part of the TFT.
Bit 24	**Reset**	**Reset.** Resets all the output signals going to flat panel including FPLINE, FPFRAME, DE, and RGB. It also forces all the internal logic to go into IDLE state. PWM signal is not affected by this signal.
Bits 23-16	**blankB**	This register represents the 8-bit value of the blue component of a blank pixel. This value is used to put blank pixels during image expansion in text mode. Note this value should correspond to the back ground colour of the current text mode.

Bit Number	Mnemonic	Description
Bits 15-8	**blankG**	This register represents the 8-bit value of the green component of a blank pixel. This value is used to put blank pixels during image expansion in text mode. Note this value should correspond to the back ground colour of the current text mode.
Bits 7-0	**blankR**	This register represents the 8-bit value of the red component of a blank pixel. This value is used to put blank pixels during image expansion in text mode. Note this value should correspond to the back ground colour of the current text mode.

Programming Notes:

To avoid damage to the flat panel, be sure to follow the directions for switching on the TFT interface as per the TFT specification being used. Normally after powering up the system, all interface signals should be low, so bit 0 should be 1. First the digital part of the flat panel should be activated, so set bit 25 as 0. After about 50ms, enable the digital interface signals by setting bit 24 to 0. After this, wait another 50ms and then enable Vee or the analog power to the TFT. When switching off the power, follow the same sequence but in the reverse order.

17.4.7. PWM CONTROL

Access = 0x084CC018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
Default value after reset =															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv				Clock Divide Value				Duty Cycle							
Default value after reset =															

Bit Number	Mnemonic	Description
Bits 31-12	**Rsv**	Reserved
Bits 11-8	**CDV**	**Clock Divide Value.** The PWM clock is generated from PCI clock divided by CDV. When CDV equals 0, the division factor is 16.
Bits 7-0	**DC**	**Duty Cycle.** The period for which the PWM signal will be active. If this value is x then the duty cycle of the PWM signal will be x/255.

Programming Notes:

The divided clock is used as the base clock for the 8 bit PWM counter. The PWM signal remains high as long as the count is less than the value specified in bits 7 to 0. To set the PWM signal to permanently high, the duty cycle should be 255. Similarly a duty cycle of zero will permanently disable the PWM signal. The formula for the PWM frequency output is
PWM_frequency = PCI_frequency / (CDV * 255) if CDV > 0

PWM_frequency = PCI_frequency / (16*255) if CDV =0

Example with PCI clock at 30 MHz :

CDV	PWM_clock (kHz)
0 (=16)	7.46
1	116.82
8	14.71
15	7.81

18. PCMCIA CONTROLLER

18.1. OVERVIEW

The Personal Computer Memory Card International Association (PCMCIA) Controller provides a single channel PC Card Interface Controller, 82365SL™ compatible, with an EXCA™ interface. The PCMCIA design is based on the PCMCIA 2.0/JEIDA 4.1 standard and provides an open standard system interface for PC Cards at the hardware and data interchange level.

18.2. INTERFACE REGISTERS

A set of interface registers is provided for PC Card Socket

The interface decoding of these registers is described in the PC Card Interface. The detailed functions of the interface registers are described in the Register Description. The following sections outline the functionality of the interface registers.

18.2.1. GENERAL SETUP REGISTERS

These registers perform the following functions:

1) Identification and Revision information for the socket.

2) Status of the Interface.

3) Power and RESETDRV Control.

4) Card Status Change.

5) Address Window Enable.

6) Card Detect and General Control Register.

7) Global Control Register.

18.2.2. INTERRUPT REGISTERS

These registers perform the following functions:

1) Interrupt and General Control.

2) Card Status Change Interrupt Configuration.

18.2.3. I/O REGISTERS

These registers perform the following functions:

1) I/O Control.

2) I/O Address window 0-1 Start/Stop Low Byte.

3) I/O Address window 0-1 Start/Stop High Byte.

18.2.4. MEMORY REGISTERS

These registers perform the following functions:

1) System Memory Address Window 0-4 Start/Stop Low Byte.

2) System Memory Address Window 0-4 Start/Stop High Byte.

3) Card Memory Index Address Window 0-4 Low Byte.

4) Card Memory Index Address Window 0-4 High Byte.

18.2.5. INTERRUPT STEERING

The PC Card Interface has 1 IRQ line for both the status and PCCard Interrupt.

Due to interrupt routing limitations for the PCMCIA.

18.3. MEMORY CONTROL

18.3.1. PC CARD MEMORY ADDRESSING

The PCMCIA enables the System User to map portions of the 64 Mbyte (26-bit address), (common and/or attribute), memory spaces on the PC Card into the smaller 16 Mbyte (24-bit address), system (ISA) address space. The mapping functions provided will allow for the system side address space to be extended to the full 64 Mbyte PC Card capability.

The PCMCIA has five independently enabled and controlled system memory address mapping windows. Each of these windows can be set to map either into the common or attribute memory space of the PC Card, (Common/Attribute Memory Address Mapping). Every system memory window has fully independent control of the memory data bus width, system bus wait states, software write protect, and enable.

System Memory windows start and stop on any 4 Kbyte boundary of the ISA system memory above 64 Kbytes. The PCMCIA will automatically inhibit any mapping of a system memory window below 64 Kbyte in the ISA system address. This is due to the fact that the first 64 K addresses are reserved for ISA I/O, and also in order to resolve conflicts in accesses to I/O PC Cards that contain memory.

In order to open any of the five windows, system memory start address, system memory stop address, and the PC Card memory offset are initialised. Accesses to the PC Card memory will occur when the following conditions have been satisfied:

1) The system memory address mapping window is enabled.

2) The ISA system memory address is greater than or equal to the mapping start register.

A23:12 (high and low byte).

3) The ISA system memory address is less than or equal to the mapping stop register.

A23:12 (high and low byte).

When the above is satisfied PCMCIA will add the PC Card memory offset address to the ISA system address in order to generate the PC Card address.

All windows can be configured independently. Alternatively, they can be grouped together in order to perform special memory mapping requirements, such as for example, LIM/EMS (Lotus-Intel-Microsoft/Extended Memory Specification), or XIP (Execute in Place).

The PC Card memory offset allows for either positive or negative (2's complement) offset from the ISA system address. The PCMCIA does not perform any checks for any window whose size and offset cause it to wrap round from the last PC Card address to the first. The system software has total responsibility for preventing address wrap round effects.

18.4. COMMON/ATTRIBUTE MEMORY ADDRESS MAPPING

As stated in PC Card Memory Addressing, Common/Attribute Memory on the PC Card can be accessed through any of the five system memory address mapping windows. This is achieved by setting the "REG active bit" in the card memory offset address register (high byte), (0 = common memory, 1 = attribute memory). There are no restrictions for the mapping of the system memory window to common/attribute memory from any ISA address to any PC Card address.

Multiple system memory address mapping windows to different common/attribute memory address spaces can be opened simultaneously. All windows can be independently configured for wait states, software write protect and data width.

18.5. MEMORY PAGING

The PCMCIA supports the paging of system memory by the use of multiple system memory address mapping windows. When using LIM or XIP, software must assign a window to each page required to support the LIM/XIP function. It is the responsibility of the software to set up the system memory address mapping windows into one contiguous system address space, and to ensure that each window controls a single page in the PC Card memory. In these conditions only the PC Card memory offset address value needs to be altered to change the page pointer and mapping.

18.6. I/O CONTROL

The PCMCIA method of I/O addressing is similar to that described in Memory Control. In this case the System User can map portions of the 0 to 64 Kbyte I/O address range of a PC Card into the smaller 768 byte I/O system (ISA) address space. The PC Card can request certain common I/O address locations or alternatively only the size of the space required. In the case when only the size of I/O space is requested, the system is able to locate the PC Card anywhere in the 64 Kbyte system I/O address space.

The PC Card decodes the CE#2, CE#1, IORD# and IOWR# signals in order to respond to an I/O access.

The PCMCIA has two independently enabled and controlled I/O address windows, each of which define a 16-Bit address (1 byte resolution). This feature allows for two non-contiguous I/O address windows to be available for each PCMCIA socket. Each of these windows has independent control of I/O data bus width, zero wait state system access and generation of IOCS16#.

An I/O PC Card can be accessed providing that the following conditions are satisfied:

1 The I/O address window is enabled.

2 The ISA system address SA[15:0] is greater than or equal to the I/O address start register (high and low byte).

3 The ISA system address SA[15:0] is less than or equal to the I/O address stop register (high and low byte).

4 The access is not a DMA transfer (AEN = ‘0’ to access the I/O PC Card).

System software is required to be responsible for accounting for each I/O address range assigned to a particular PC Card. Card power consumption can be reduced by the reservation of a particular I/O address range for each PC Card, this is due to the fact that only one PC Card is enabled during each I/O access.

As there is no indirect offset addressing within the PCMCIA, the system's I/O address must be within the PCMCIA I/O window in order to generate the correct I/O cycle to the PC Card. The I/O addressing bits [15:0] of the PC Cards are directly derived from SA[15:0] with bits [25:16] driven “0".

The PCMCIA directly maps the system I/O address space to the PC Card I/O ports to a single byte resolution. Each PC Card is guaranteed a reserved system I/O space, and an I/O cycle will be generated to the PC Card within the assigned space. This means that the PCMCIA does not rely on the PC Card to decode the I/O address space and respond with the acknowledge signal. However, for systems that generate card enables to the PC Card over a wide range of I/O address space, and rely on the PC Card acknowledge signals to enable any data transreceivers between the PC Slot and the system data bus, the PCMCIA Input Acknowledge (PACK#) will be required. On a read from the PC Card with an address window enabled, the PCMCIA will qualify the data transreceiver direction line and the card enables with a valid access to the PC Card I/O address space. The PCMCIA will not allow overlapping I/O address windows to be enabled concurrently.

18.7. PCMCIA CONTROL

18.7.1. PCMCIA CARD STATUS

The status of the PC Card is accessible through the interface status register. The status information in this register includes PC Card detection, memory write protect status, battery voltage detect, PC Card power, and ready/busy. Changes in PC Card status can cause a card status change interrupt, (for example when a PC Card is inserted or removed). The source of the interrupt is configurable.

18.7.2. CONTROL/STATUS SIGNAL MULTIPLEXERS

PC Card signals have differing designations depending on whether the PC Card is memory or I/O. The PCMCIA contains multiplexers to redirect the appropriate signals based upon the condition of the PC Card type bit in the interrupt and general control register.

18.7.3. CONFIGURATION REGISTERS

The PCMCIA provides a register containing interface identity and revision information for the socket.

18.7.4. POWER MANAGEMENT

The PCMCIA provides independent power management control signals for each PC Card socket. Socket power management is controlled by programming the POWER and RESETDRV control register.

The actual power buffers are required to be added externally by the System Designer, refer to (2.9).

The PCMCIA automatically enters into a lower power consumption state when the memory and I/O windows are disabled and the sockets are empty. The lowest power consumption level can be achieved by setting the PCMCIA into CS# controlled power down mode. This mode is entered by disabling all the I/O/ Memory windows, output buffers, enabling the power down mode, and driving CS# high.

During the CS# power controlled down mode the PCMCIA will still be powered up and the contents of the internal registers will be maintained. Interrupt requests (IRQs) can still be generated to the Host System by the PCMCIA from either card status change, or PC Card interrupt requests. Additionally RI_OUT# can still be set to route either RI_IN#, or Card detect change. Furthermore INTR# can also be set to route either card status change interrupts or PC Card interrupts.

The PCMCIA provides a unique feature to support host system suspend/resume operations. When the host system enters suspend mode the PCMCIA still maintains the ability to route the RI# signals from both sockets to the RI_OUT# signal as the resume indication to the host system.

18.7.5. PC CARD INTERFACE

The PCMCIA directly supports one PC Card Sockets (A). The PC Card socket interface has its own complement of memory address mapping, I/O address mapping, configuration and status registers. Additionally a set of ID and revision registers is provided.

All PCMCIA control registers are byte wide and are accessed using the following indirect indexing scheme. Two ISA I/O addresses are used to access the PCMCIA control registers. The first ISA I/O address is the index register and the second ISA I/O address the data register. The PC Card socket can have up to 64 indirectly addressed registers, thus allowing support of the two separate PC Card Sockets using only 2 ISA I/O addresses.

18.7.6. INTERFACE DECODE LOGIC

The interface decoding logic decodes the ISA I/O addresses 3E0h and 3E1h as the addresses of the PCMCIA index and data registers respectively. To access one of the PCMCIA registers the system must first write the index value to the index register, and then either read or write to the data register. Both the index and data registers are read/write. Providing the index register contains a valid index, the PCMCIA will respond to a data register read/write operation, or to an index register read operation.

A mismatch in the address decode will cause a clearing of all 4 of the index registers in the PCMCIA (see Table 18-1.).

Table 18-1. Index Register Mapping

Socket A	Base Address	Index Range
Slot 0	3E0h	00h to 3Fh

18.8. EXTERNAL CONNECTIONS

The external connections required by the System Designer are illustrated in Fig. 2.9. The PC Card data (SD[15:0]) and address (SA[11:0]) bus is electrically isolated from the internal ISA bus by two sets of transceivers and buffers. The PCMCIA provides control signals to enable the transfer of odd or even data bytes, (CE#1 and CE#2 respectively) between the PC Card and the ISA bus via transceivers. The system address is electrically isolated by a buffer. Power Control signals are provided from the PCMCIA and these control via isolating circuitry VCC/VPP to each card socket.

Note: I/O PC Card Functions in the following table are listed directly below the Memory PC Card functions and are designated in ( ).

18.9. PCMCIA GENERAL SET UP REGISTER DESCRIPTION

18.9.1. IDENTIFICATION AND REVISION REGISTER (R)

This register is used by the system software to determine the type of PC Cards supported, and also to identify which PCMCIA version is present. The system software reads this register and then compares the result value against existing revision numbers.

Id_Rev Access = 0xCF8h/0xCFCh Regoffset = 0x00h

7	6	5	4	3	2	1	0
PCMIT		Rsv		PCMREV			
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-6	**PCMIT**	**PCMCIA Interface Type.** These bits indicate the type of PC Cards supported by the PCMCIA at the particular socket. These bits do not identify the type of card that is present at the socket (see Table 18-2.) **Note:** These bits will be read back as 10.
Bits 5-4	**Rsv**	**Reserved.** Bits 5 and 4 will be read back as zero.
Bits 3-0	**PCMREV**	**PCMCIA Revision.** Bits 3 to 0 indicate the current revision level of the PCMCIA. Bits 3 to 0 will be read back as 0011.

Table 18-2. PCMCIA Interface Type

Bit 7	Bit 6	Interface
0	0	I/O Only
0	1	Memory Only
1	0	Memory & I/O
1	1	Reserved

18.9.2. INTERFACE STATUS REGISTER (R)

This register provides the current status of the PC Card socket interface signals.

Active BSY# SPKR# STSCHG#

Int_Stat Access = 0xCF8h/0xCFCh Regoffset = 0x01h

7	6	5	4	3	2	1	0
GPI#	PCCPA	RDBSY	MWP	CD12		BVD12	
Default value after reset =							

Bit Number	Mnemonic	Description
Bit 7	**GPI#**	**GPI#.** This bit is hardwired to 1.
Bit 6	**PCCPA**	**PC Card Power Active.** This bit indicates the current power status of the socket. If bit 6 = '0', power to the socket is turned off, (VCC ,VPP1 and VPP2 are no actively driven). When bit 6 = '1', power is provided to the socket (VCC = 5V, and VPP1 and VPP2 are set according to bits 3-0 in the power control register).
Bit 5	**RDBSY**	**Ready/BSY#.** This bit indicates the ready condition of the PC Memory Card. If this bit = 1, the PC Card is ready to accept a new data transfer. When this bit = 0, the PC Card is busy processing a previous command, or alternatively performing initialization for an I/O Card. The default state of bit 5 = 1.
Bit 4	**MWP**	**Memory Write Protect.** This bit gives the logic level of the WP pin. If WP pin = '0', bit 4 = '0'. If WP pin = '1', bit 4 = '1'. Memory write access to the slot will not be blocked unless the write protect bit in the associated Card Memory Index Address Register High byte is set to a one.
Bits 3-2	**CD12**	**Card Detect 2 and 1.** These bits indicate whether a card is present at the socket and fully seated. If CD#1 = 0, bit 2 = '1'. If CD#1 = 1, bit 2 = '0'. If CD#2 = 0, bit 3 = '1'. If CD#2 = 1, bit 3 = '0'. They are set to zero if the PC Card interface are inactive.
Bits 1-0	**BVD12**	**Battery Voltage Detect 2 and 1.** Table 18-3. shows the possible battery states, as indicated by bits 1and 0of this register. For I/O PC Card, bit 0 indicates the current status of the STSCHG#/RI# signal from the PC Card, while bit 1 indicates the current state of SPKR# from the PC Card.

Table 18-3. Battery Voltage Detect

Bit 1	Bit 0	Status
0	x	Battery Dead
1	0	Warning
1	1	Battery Good

18.9.3. POWER AND RESETDRV CONTROL REGISTER

This register provides both the control of PC Card Power and the resetting of the PCMCIA registers.

A RESETDRV clears all bits in this register, unless the RESETDRV is a result of a Host System resume (PWRGOOD='1') and the disable resume RESETDRV bit is set to a one. (Output Enable (D6) should not be set until the register has been previously written setting PC Card Power Enable (D4)).

PRD_Cont Access = 0xCF8h/0xCFCh Regoffset = 0x02h

7	6	5	4	3	2	1	0
OE	RDR	APSE	PCPE	PCVP2		PCVP1	
Default value after reset =							

Bit Number	Mnemonic	Description
Bit 7	**OE**	**Output Enable.** If Bit 7 is set to zero, the following PC Card Outputs are tri-stated, and the ENABLE# pin to the corresponding socket is inactive: CA[25:12], CE#1, CE#2, IORD#, IOWR#, OE#, REG#, RESET, WE#. The functionality of the Output Enable bit is shown in Table 18-8. This bit should always be set to zero when a slot is not powered.
Bit 6	**RDR**	**RESETDRV Disable Resume.** This bit specifies the action of RESETDRV signal under resume (PWRGOOD='1'). See Table 18-4.
Bit 5	**APSE**	**Auto Power Switch Enable.** Automatic socket power switching based on card detects is disabled when this bit is set to zero. Conversely if the bit is set to a one then automatic power switching is enabled. The automatic socket power switching function controls the VCC_EN# and VPP_ENX output pins. The PCMCIA provides each socket with 5 power control pins. These are physically used to control the power supplies (VCC, VPP2 and VPP1) to the PC Cards. See Table 18-5.
Bit 4	**PCPE**	**PC Card Power Enable.** When this bit is set to zero, all power to the PC Card is disabled. However when this bit is set to a one VCC_EN# = 0, VCC= 5V, and VPP1/VPP2 are set in accordance with the states of bits [3:0] in this register.
Bits 3-2	**PCVP2**	**PC Card VPP 2 Power Control.** Bits 3 and 2 control the VPP2_EN1, VPP2_EN0 output pins of PCMCIA as shown in Table 18-6. **Note:** The 11 combination of Bits 2 and 3 is reserved and should not be used.
Bits 1-0	**PCVP1**	**PC Card VPP1 Power Control.** Bits 1 and 0 control the VPP1_EN1, VPP1_EN0 output pins of PCMCIA as shown in Table 18-7. **Note:** The 11 combination of Bits 1 and 0 is reserved and should not be used.

Table 18-4. Signal RESETDRV Action

Disable RESETDRV	Signal PWRGOOD	Signal RESETDRV Action
0 or 1	0	System reset. Resetable registers are reset.
0	1	System resume. Resetable registers are reset.
1	1	System resume. Resetable registers are not reset.

Table 18-5. Power Control Pins

Control Bit	Name	Description
0	VPP1_EN0	VPP 1 control, bit 0
1	VPP1_EN1	VPP 1 control, bit 1
2	VPP2_EN0	VPP 2 control, bit 0
3	VPP2_EN1	VPP 2 control, bit 1
4	VCC_EN#	Master Enable

Table 18-6. VPP2 Outputs

Bit 4	Bit 3	Bit 2	VPP2_EN1	VPP2_EN0	VCC_EN#
1	0	0	0	0	0
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	0	0	0
0	x	x	0	0	1

Table 18-7. VPP1 Outputs

Bit 4	Bit 1	Bit 0	VPP1_EN1	VPP1_EN0	VCC_EN#
1	0	0	0	0	0
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	0	0	0
0	x	x	0	0	1

Programming notes:

Slot Power Control

A PC Card is considered to be present (detected) in the socket when both CD#2 and CD#1 = '0'. The PC Card Power Active bit of the Interface Register indicates the current power status of the socket. When this bit is zero, power to the socket is turned off (VCC_EN#, VPP1_EN0, VPP1_EN1, VPP2_EN0, and VPP2_EN1 are all inactive). However, when the bit is set to a one, power is provided to the socket. (VCC_EN# = 0V and VPP1 and VPP2 are set according to bits 3-0 in the power control register). Table 18-8. summarises the slot power control function.

Note: PC Card Power Active = 0 means VPPX_EN#, VCC_EN#, and NVCC_EN pins are active to provide power supply to VCC, VPP1 and VPP2.

The power control circuitry only switches the applicable voltages and does not provide voltage generation. For compliance with IEC 950 and UL 1950 It is recommended that a 1A, 125V fuse be installed on each PCMCIA socket voltage line.

.

Table 18-8. Slot Power Control

Power Control Register			PC Card Pins		Logic Outputs	Interface Status Register
Output Enable (D7)	**PC Card Power Enable (D4)**	**Auto Power Switch Enable (D5)**	**CD#1**	**0CD#2**		**PC Card Power Active**
X	0	X	X	X	OFF	0
0	1	0	0	0	OFF	1
1	1	0	0	0	ON	1
X	1	0	X	1	OFF	1
X	1	0	1	X	OFF	1
0	1	1	0	0	OFF	1
1	1	1	0	0	ON	1
X	1	1	X	1	OFF	0
X	1	1	1	X	OFF	0

18.9.4. CARD STATUS CHANGE REGISTER

CSC Access = 0xCF8h/0xCFCh Regoffset = 0x04h

7	6	5	4	3	2	1	0
Rsv			Rsv	CDC	RC	BW	BD
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-5	**Rsv**	**Reserved.** These bits always read zero.
Bit 4	**Rsv**	**Reserved.** This bit is hardwired to 1.
Bit 3	**CDC**	**Card Detect Change.** Bit 3 is set to a one when a change has been detected on either CD#1 or CD#2.
Bit 2	**RC**	**Ready Change.** Bit 2 is set to a one when a low to high transition has been detected on Ready/NBUSY, this indicates that the memory PC Card is ready to accept a new data transfer. For I/O PC Cards this bit reads zero.
Bit 1	**BW**	**Battery Warning.** When a battery warning condition has been detected this bit is set to a one. For I/O PC Cards this bit reads zero.
Bit 0	**BD**	**Battery Dead (STSCHG#).** Bit 0 is set to a one when a battery dead condition has been detected on Memory PC Cards. **Note:** For I/O PC Cards this bit is set to a one if the ring indicate enable bit is set to zero (Interrupt and General Control Register) and the (STSCHG#/RI#) signal from the I/O PC Card has been pulled low. The system software then has to read the status change register in the PC Card to determine the cause of the status change signal (STSSCHG#). This bit also reads zero for I/O PC Cards if the ring enable bit is set to a one.

Programming notes:

This register contains the status of the sources for the card status change interrupts. These sources can be enabled to generate a card status change interrupt by setting the appropriate bit in the Card Status Change Interrupt Configuration Register. The bits in this register (Card Status Change) will be read back as '0' when the Card Status Enable bits are set to zero, in the Card Status Change Interrupt Configuration Register, for the various sources of the card status change interrupts.

Where the Explicit Write Back Card Status Change Acknowledge bit is set in the Global Control Register, the acknowledgement of the card status change interrupt sources is achieved by writing a one back to the appropriate bit in the Card Status Change Register that was read as a one. Once acknowledged, that particular bit in the Card Status Change Register will be read back as '0'. The interrupt caused by a card status change (if enabled on a system IRQ line) will be active until all of the bits in this register are at zero.

Should the Explicit Write Back Card Status Acknowledge bit not be set, the card status change interrupt, when enabled on a system IRQ line, will remain active until the Card Status Change Register is read. This read operation will reset all the bits in the Card Status Change Register.

Where two or more card status change interrupts are pending, and a card status change interrupt occurs while serving one source of card status change, the PCMCIA will not generate a second interrupt pulse.

Therefore, in explicit write back acknowledge mode, the Host System interrupt service routine must acknowledge each Card Status Change Interrupt source by writing 1's to the respective bits in the Card Status Change Register. In the standard acknowledge mode, the software interrupt service routine must first read the Card Status Register to store all the card status change sources and then service them in turn.

In both of the above modes the Interrupt Service Routine needs the Card Status Change Register in order to be sure that all interrupt requests are serviced before issuing any service routines.

A RESTDRV clears all the bits in this register, unless RESETDRV is the result of a Host System resume (PWRGOOD = '1'), and Disable RESETDRV = '1' (Power and RESETDRV control register).

Note: In the following sections, bit descriptions in ( )'s indicate valid signals after the interface is configured for I/O Cards.

18.9.5. ADDRESS WINDOW ENABLE REGISTER

This register controls the enabling of the memory and I/O mapping windows to the PC Card memory or I/O space. RESETDRV clears all bits in this register, unless the RESETDRV is a result of a Host System resume (PWRGOOD='1') and disable resume RESETDRV is set to a one (Power and RESETDRV control register).

In order for the CS# controlled power-down mode to function properly, all the memory and I/O window enable bits in this register need to be set to zero before the PCMCIA enters the power-down mode.

AW_En Access = 0xCF8h/0xCFCh Regoffset = 0x06h

7	6	5	4	3	2	1	0
IOW1	IOW0	MRM16D	MW4E	MW3E	MWE2	MWE1	MWE0
Default value after reset =							

Bit Number	Mnemonic	Description
Bit 7	**IOW1**	**I/O Window 1 Enable.** In the condition where bit 7 = '0', an I/O access within the I/O address 1 window will inhibit card enable signals to the PC Card. Conversely if bit 7 = '1', an I/O access within the I/O address 1 window will generate card enables to the PC Card. (The start and stop register pairs must all be set to the desired window values before setting this bit to one).
Bit 6	**IOW0**	**I/O Window 0 Enable.** In the condition where bit 6 = '0', an I/O access within the I/O address 0 window will inhibit card enable signals to the PC Card. Conversely if bit 6 = '1', an I/O access within the I/O address 1 window will generate card enables to the PC Card. (The start and stop register pairs must all be set to the desired window values before setting this bit to one).
Bit 5	**MRM16D**	**MRMCS16# Decode.** When bit 5 is set to zero, MEMCS16# is generated from a decode of the system address lines SA[23:17] only. Therefore at a minimum a 128K block of system (ISA) memory address space is set aside as 16-bit memory only. If bit 5 = '1' MEMCS16# is generated from a decode of the system address lines SA[23:12] (4K block). Full line address decode should be used when decoding in the first 128K block of address space.
Bit 4	**MW4E**	**Memory Window 4 Enable.** When bit 4 = '0', a memory access within the system memory address 4 window will inhibit the card enable signals to the PC Card. If however bit 4 = '1', a memory access within the system memory address 4 window will generate the card enables to the PC Card. The start, stop and offset register pairs must all be initialised before setting bit 4 to a one. When bit 4 = '1' the PC Card address is generated for valid window addresses.
Bit 3	**MW3E**	**Memory Window 3 Enable.** When bit 3 = '0', a memory access within the system memory address 3 window will inhibit the card enable signals to the PC Card. If however bit 3 = '1', a memory access within the system memory address 3 window will generate the card enables to the PC Card. The start, stop and offset register pairs must all be initialised before setting bit 3 to a one. When bit 3 = '1' the PC Card address is generated for valid window addresses.

Bit 2	**MW2E**	**Memory Window 2 Enable.** When bit 2 = '0', a memory access within the system memory address 2 window will inhibit the card enable signals to the PC Card. If however bit 2 = '1', a memory access within the system memory address 2 window will generate the card enables to the PC Card. The start, stop and offset register pairs must all be initialised before setting bit 2 to a one. When bit 2 = '1' the PC Card address is generated for valid window addresses.
Bit 1	**MW1E**	**Memory Window 1 Enable.** When bit 1 = '0', a memory access within the system memory address 1 window will inhibit the card enable signals to the PC Card. If however bit 1 = '1', a memory access within the system memory address 1 window will generate the card enables to the PC Card. The start, stop and offset register pairs must all be initialised before setting bit 1 to a one. When bit 1 = '1' the PC Card address is generated for valid window addresses.
Bit 0	**MW0E**	**Memory Window 0 Enable.** When bit 0 = '0', a memory access within the system memory address 0 window will inhibit the card enable signals to the PC Card. If however bit 0 = '1', a memory access within the system memory address 0 window will generate the card enables to the PC Card. The start, stop and offset register pairs must all be initialised before setting bit 0 to a one. When bit 0 = '1' the PC Card address is generated for valid window addresses.

18.9.6. CARD DETECT AND GENERAL CONTROL REGISTER

A RESETDRV clears all bits in this register, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = '1' in the slot Power and RESETDRV Control Register.

C_Cont Access = 0xCF8h/0xCFCh Regoffset = 0x16h

7	6	5	4	3	2	1	0
Rsv		SCDI	CDRE	GPITC	Rsv	CRE	16MDI
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-6	**Rsv**	**Reserved.** These bits should not be used.
Bit 5	**SCDI**	**Software Card Detect Interrupt.** In the condition where the Card Detect Enable Bit = '1' (Card Status Change Interrupt Configuration Register), then writing a '1' to bit 5 will cause a card detect card status change interrupt for the associated slot. The functionality and acknowledgement of this software interrupt will work in the same way as the hardware generated interrupt. **Note 1:** The functionality of the hardware card detect card status change interrupt will not be affected. The previous state of the CD#1 and CD#2 inputs will be latched such that while a S/W card detect card status change interrupt occurs and is being serviced, any new changes of state of CD#1 and CD#2 will cause an H/W card detect card status change interrupt to be generated. If the Card Detect Enable Bit = 0 (Card Status Change Interrupt Configuration Register), then writing a 1 to the S/W Card Detect Interrupt bit has no effect. **Note 2:** The S/W Card Detect Interrupt bit will always read back as a '0'.
Bit 4	**CDRE**	**Card Detect Resume Enable.** The default state of this bit = '0'. In the condition where bit 4 = '1', then oCE# a card detect change has been detected on the CD#1 and CD#2 inputs, the RI_OUT# output will change state from high to low and the Card Detect Change bit (Card Status Change Register) will be set to a '1'. The RI_OUT# output will remain low until either a read or a write of 1 to the Card Detect Change bit (Card Status Change Register), (acknowledge cycle), causes the Card Detect Change bit to be cleared and RI_OUT# output to change state from low to high. The Card Detect Enable bit must be set in the Card Status Change Interrupt Configuration Register in order to generate RI_OUT#. In the condition where the card status change is routed to either INTR# or any IRQ, the setting of Card Detect Resume Enable bit to a one will prevent INTR# and IRQ from going active as a result of card status change. OCE# the resume software has detected a card detect change interrupt from RI_OUT# (Card Status Change Register read), the software should initiate a software card detect change so that the card detect change condition will generate an active interrupt on IRQ or INTR# (which one being dependent on the active configuration). When bit 4 is set to '0', then the card detect resume functionality is disabled. Therefore RI_OUT will not go low due to a card detect change. The RI_OUT# output will be the logical AND of all the active low sources for ring indicate output including the RI# inputs from slot A and slot B, together with the card detect changes on CD#1, CD#2 from both slots.

Bit 3	**GPITC**	**GPI Transition Control.** The default state of bit 3 = '0'. The setting of the General Purpose Input (GPI) Enable bit to a '1' will enable a card status change interrupt when the GPI# input changes state from a high to a low, providing bit 3 (GPI Transition Control) = '0'. If the GPI Enable bit is set to a one and bit 3 is set to a one, then oCE# the GPI# input changes state from low to high a card status change interrupt will be generated. When GPI# is used for card ejection/insertion pending events, transition control allows for either of these events, (not both), to cause an interrupt.
Bit 2	**Rsv**	**Reserved.** This bit is hardwired to 1.
Bit 1	**CRE**	**Configuration Reset Enable.** The default state of bit 1 = '0'. There is one bit for each slot. If bit 1 is set to a '0', the configuration register reset function based on card detects is disabled. When bit 1 is set to a '1', in the condition where both the CD#1 and CD#2 inputs for a particular slot go high, a reset pulse will be generated. This pulse will reset the configuration registers for that particular slot to their default states ('0's). Note: Bit 1 is required to be set by the card detect change interrupt service routine only when a PC Card is inserted, and reset when the card is removed. It is also a requirement to enable the card detect card status change interrupt if the card detect configuration reset function is to be used in the CS# power down mode. Table 18-9. shows a list of registers that will be reset to zero when the Configuration Reset Enable is set to a 1 and both the CD#1 and CD2 inputs go high.
Bit 0	**16MDI**	**16-Bit Memory Delay Inhibit.** The default state of bit 0 = '0'. This is not programmable on a per window basis.When bit 0 is set to '0' and a system memory window is set up to be 16-bit, (Data Size Bit = '1' in the System Memory Address Mapping Start High Byte Register), the falling edges of the control strobes EW# and OE# for the corresponding slot will be delayed synchronously by SYSCLK. These falling edges will be generated from the first falling edge of SYSCLK occurring after the falling edge of MEMW# or MEMR#, and will be also be gated by a valid system memory window decode. The rising edge of the control strobes will be generated from the rising edge of MEMW# or MEMR#. In the condition where bit 0 is set to a '1' and the system memory window is set up to be 16-bit, the control strobes WE# and OE# for the corresponding slot will not be synchronously delayed by SYSCLK.

Table 18-9. Slot Registers set to Zero

Slot	Description
03h	Interrupt and General Control **(Except 'INTR# Enable' bit)**
06h	Address Window Enable **(Except 'MEMCS#16 Decode A23:12-bit)**
07h	I/O Control
08h	I/O Address 0 Start Low Byte
09h	I/O Address 0 Start High Byte
0Ah	I/0 Address 0 Stop Low Byte
0Bh	I/O Address 0 Stop High Byte
0Ch	I/O Address 1 Start Low Byte
0Dh	I/O Address 1Start High Byte
0Eh	I/O Address 1 Stop Low Byte
0Fh	I/O Address 1 Stop High Byte
10h	System Memory Address 0 Mapping Start Low Byte
11h	System Memory Address 0 Mapping Start High Byte
12h	System Memory Address 0 Mapping Stop Low Byte
13h	System Memory Address 0 Mapping Stop High Byte
14h	Card Memory Index Address 0 Low Byte
15h	Card Memory Index Address 0 High Byte
18h	System Memory Address 1 Mapping Start Low Byte
19h	System Memory Address 1 Mapping Start High Byte
1Ah	System Memory Address 1 Mapping Stop Low Byte
1Bh	System Memory Address 1 Mapping Stop High Byte
1Ch	Card Memory Offset Address 1 Low Byte
1Dh	Card Memory Offset Address 1 High Byte
20h	System Memory Address 2 Mapping Start Low Byte
21h	System Memory Address 2 Mapping Start High Byte
22h	System Memory Address 2 Mapping Stop Low Byte
23h	System Memory Address 2 Mapping Stop High Byte
24h	Card Memory Offset Address 2 Low Byte
25h	Card Memory Offset Address 2 High Byte
28h	System Memory Address 3 Mapping Start Low Byte
29h	System Memory Address 3 Mapping Start High Byte
2Ah	System Memory Address 3 Mapping Stop Low Byte
2Bh	System Memory Address 3 Mapping Stop High Byte
2Ch	Card Memory Offset Address 3 Low Byte
2Dh	Card Memory Offset Address 3 High Byte
30h	System Memory Address 4 Mapping Start Low Byte
31h	System Memory Address 4 Mapping Start High Byte
32h	System Memory Address 4 Mapping Stop Low Byte
33h	System Memory Address 4 Mapping Stop High Byte
34h	Card Memory Offset Address 4 Low Byte
35h	Card Memory Offset Address 4 High Byte

18.9.7. GLOBAL CONTROL REGISTER

A RESETDRV clears all bits in this register, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = 1' in the slot Power and RESETDRV Control Register.

Global Access = 0xCF8h/0xCFCh Regoffset = 0x1Eh

7	6	5	4	3	2	1	0
Rsv				IRQ14PME	EWB	LMIE	PD
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-4	**Rsv**	**Reserved.** These bits should not be used.
Bit 3	**IRQ14PME**	**IRQ14 Pulse Mode Enable.** When bit 3 = '1' and bit 1 = '0' (level mode interrupt enable), the PCMCIA is enabled to use IRQ14 to support the PC Card with a pulse mode interrupt on IREQ#. In this condition other IRQ's will still support edge-trigger interrupts from either card status change interrupts, or PC Card IREQ#. However if bit '1' = 1 (level mode interrupt enable), then bit 3 has no effect on IRQ14.
Bit 2	**EWB**	**Explicit Write Back CSC Ack.** Setting this bit = '1', will require an explicit write of a one to the Card Status Change Register bit which indicates an interrupting condition. When this bit = '0' (default state), the card status change interrupt is acknowledged by reading the Card Status Change Register, and the register bits are cleared upon a read.

Bit 1	**LMIE**	**Level Mode Interrupt Enable.** When Bit 1 = '1', then all the IRQ outputs are configured to be active low level mode interrupts. In this mode, an IRQ will remain at the output logic state until there is either a card status change interrupt or an active I/O card interrupt is steered to that particular IRQ, at which time the IRQ output will go low. In the case of the interrupt being caused by an IREQ# active (low) from a PC Card, IRQ will remain low until IREQ# becomes inactive, then IRQX will be de-asserted. For an interrupt caused by the card status change, IRQ will remain low until the interrupt is acknowledged (service). OCE# serviced the IRQ output will go from a low to the output logic state. When Bit 1 = '0' (Default state), the IRQ outputs will be configured to be low to high edge triggered interrupts. In this mode the IRQs will remain at the output logic state until a particular IRQ is enabled which will then cause the IRQ output to go low. The output will remain low until a card status change interrupt or I/O card interrupt occurs to force the IRQ output high. For the interrupt caused by the IREQ# active (low) from a PC Card IRQX will remain high until IREQ# becomes inactive, then IRQ will be low again. In the state where the interrupt is caused by the card status change, it will remain high until the interrupt is acknowledged (serviced) when it will be forced low again. In either of these cases, IRQs will remain low until they are disabled (interrupt and general control register). When disabled, IRQ will be at the output logic level state.
Bit 0	**PD**	**Power Down.** The PCMCIA enters the CS# controlled power down when this bit = '1'. In this case all the I/O memory windows are disabled and CS# is driven to inactive high. During CS# controlled power down, all the PCMCIA's internal registers are inaccessible, outputs are disabled and minimum power consumption level is enabled. IRQs and RI_OUT# will however still be active to monitor the card detect and RI# status for any sign of a resume indication.

18.10. INTERRUPT REGISTERS

18.10.1. INTERRUPT AND GENERAL CONTROL REGISTER

This register controls the interrupt steering for the PC Card I/O interrupt as well as providing general control of the PCMCIA. A RESETDRV clears all bits in this register, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = '1' in the slot Power and RESETDRV Control Register.

Int_Cont Access = 0xCF8h/0xCFCh Regoffset = 0x03h

7	6	5	4	3	2	1	0
RIE	PCCR	PCCT	IE#	ILS			
Default value after reset =							

Bit Number	Mnemonic	Description
Bit 7	**RIE**	**Ring Indicate Enable.** When this bit = '1' and the PC Card Type bit (5) is set to a one (I/O PC Card), (STSCHG#/RI) from the I/O PC Card is used as a ring indicator signal and is passed through to the RI_OUT# output in of the PCMCIA. When this bit = '0' and the PC Card Type bit (5) is set to a one (I/O PC Card), (STSCHG#/RI) from the I/O PC Card is used as the status change signal (STSCHG#). The current status of the signal is then available to be read from the Interface Status Register (01H), and this signal can be configured as a source for the card status change interrupt. This bit has no function when the PC Card type bit is set to zero (memory PC Card). See Table 18-10.
Bit 6	**PCCR**	**PC Card RESET#.** This is a software reset to the PC Card. Setting bit 6 = '0' activates the RESET signal to the PC Card. The RESET signal will remain active until bit 6 is set to a one.
Bit 5	**PCCT**	**PC Card Type.** A I/O PC Card is selected by setting bit 5 = '1', thus enabling the PC Card interface multiplexer for routing of PC Card I/O signals. A Memory PC Card is selected by setting bit 5 = '0'.
Bit 4	**IE#**	**INTR# Enable.** When bit 4 is set to a one it enables the card status change interrupt on the INTR# signal. However if bit 4 = '0', INTR# will always be inactive, and the card status change interrupt is steered to one of the IRQ lines in accordance# with bits 7-4 in the card status change interrupt configuration register. In CS# power down mode, INTR# will be inactive even if the bit is set and card status changes occur.
Bits 3-0	**ILS**	**IRQ Level Selection.** This selection is only applicable to I/O Cards. The following Table shows the functionality of bits 3-0 to control the re-direction of the PC Card interrupt. See Table 18-11.

.

Table 18-10. Bit 7 Function

Bit 7	Bit 5	Function
0	0	No function
0	1	STSCHG#
1	0	No function
1	1	RI# → RI_OUT#

Table 18-11. PC Card IREQ# Interrupt Steering

IRQ Bit 3	IRQ Bit 2	IRQ Bit 1	IRQ Bit 0	Function
0	0	0	0	IRQ Not Selected
0	0	0	1	*Reserved*
0	0	1	0	*Reserved*
0	0	1	1	IRQ3 Enabled
0	0	0	0	IRQ4 Enabled
0	1	0	1	IRQ5 Enabled
0	1	1	0	*Reserved*
0	1	1	1	IRQ7 Enabled
1	0	0	0	*Reserved*
1	0	0	1	IRQ10 Enabled
1	0	1	1	IRQ11 Enabled
1	1	0	0	IRQ12 Enabled
1	1	0	1	*Reserved*
1	1	1	0	IRQ14 Enabled
1	1	1	1	IRQ15 Enabled

18.10.2. CARD STATUS CHANGE INTERRUPT CONFIGURATION REGISTER

This register controls the interrupt steering of the card status change interrupt and the card status change interrupt enables. A RESETDRV clears all bits in this register, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = '1' in the slot Power and RESETDRV Control Register.

CSCIC Access = 0xCF8h/0xCFCh Regoffset = 0x05h

7	6	5	4	3	2	1	0
CSCIS				CDE	RE	BWE	BDE
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-4	**CSCIS**	**CSC Interrupt Steering.** These bits select the re-direction of the card status change interrupt if the interrupt is not selected to the output. Table 18-12. shows the routing of card status change interrupts.
Bit 3	**CDE**	**Card Detect Enable.** When bit 3 is set to a one a card status change interrupt is enabled when a change has been detected on CD#1 or CD#2. Conversely if bit 3 is set to zero this disables the generation of a card status change interrupt when CD#1, CD#2 change state.
Bit 2	**RE**	**Ready Enable.** When bit 2 = '1' a card status change interrupt is enabled when a low to high transition has been detected on Ready/NBusy. Conversely if bit 2 = '0', a card status change interrupt generation is disabled when a low to high transition has been detected on Ready/Busy#. This bit is ignored when the interface is configured for I/O PC Cards.
Bit 1	**BWE**	**Battery Warning Enable.** When bit 1 is set to a one, a card status change interrupt is enabled when a battery warning condition is detected. Conversely if bit 1 is set to zero this disables the generation of a card status change interrupt when a battery warning condition has been detected. This bit is ignored when the interface is configured for I/O PC Cards.
Bit 0	**BDE**	**Battery Dead Enable (STSCHG#).** For Memory PC Cards, the setting of Bit 0 to a one enables a card status change interrupt when a battery dead condition has been detected. For I/O PC Cards, the setting of Bit 0 to a one enables the PCMCIA to generate a card status change interrupt providing (STSCHG#/RI#) has been pulled low by the I/O PC Card and also assuming that the Ring Indicate Enable bit = ‘0’ (Interrupt and General Control Register). Setting Bit 0 = '0' disables the generation of a card status interrupt. This bit is ignored when the interface is configured for I/O PC Cards and the Ring Enable bit = ‘1’ (Interrupt and General Control Register).

.

Table 18-12. CSC Interrupt Steering

INTR# Enable Bit *	IRQ Bit 3	IRQ Bit 2	IRQ Bit 1	IRQ Bit 0	Interrupt Request Level
0	0	0	0	0	IRQ Not Selected
0	0	0	0	1	*Reserved*
0	0	0	1	0	*Reserved*
0	0	0	1	1	IRQ3 Enabled
0	0	0	0	0	IRQ4 Enabled
0	0	1	0	1	IRQ5 Enabled
0	0	1	1	0	*Reserved*
0	0	1	1	1	IRQ7 Enabled
0	1	0	0	0	*Reserved*
0	1	0	0	1	IRQ10 Enabled
0	1	0	1	1	IRQ11 Enabled
0	1	1	0	0	IRQ12 Enabled
0	1	1	0	1	*Reserved*
0	1	1	1	0	IRQ14 Enabled
0	1	1	1	1	IRQ15 Enabled
1	X	X	X	X	Card Status Change Interrupt on INTR# pin
* Within Interrupt and General Control Register.					

18.11. I/O REGISTERS

18.11.1. I/O CONTROL REGISTER

These registers indicate the I/O configuration for I/O window 0 and 1. This information is read from the PC Card card information structure. Dynamic bus sizing on a cycle by cycle basis is implemented to the PC Card interface if the source of IOCS16# is the PC Card (determined by the IOCS16# source bit). In order to be compatible with a variety of software/hardware implementations (IDE interface for example) it is necessary for the PC Card to decode two consecutive I/O addresses to determine the cycle data width.

Also in order to meet the system bus timings, this type of PC Card must decode the address lines SA[9:0] prior to the card enable signal becoming active at the interface. The card decodes the address and responds to a 16-bit cycle by enabling IOS16#. The PCMCIA qualifies IOS16# with the card enable signals in order to generate IOCS16# to the system bus.

The individual bits in these registers set the data path size and select zero wait states for the appropriate bus access. Additionally they determine the system bus signal IOCS16# as described above.

A RESETDRV clears all bits in these registers, unless RESETDRV is a result of a Host System resume (PWRGOOD = 1), and the Disable Resume RESETDRV bit = 1 in the slot Power and RESETDRV Control Register.

I/O_Cont Access = 0xCF8h/0xCFCh Regoffset = 0x07h

7	6	5	4	3	2	1	0
IOW1WS	IOW10W	IOW1IOCS		IOW0WS	IOW00W	IOW0IOCS	IOW0DS
Default value after reset =							

Bit Number	Mnemonic	Description
Bit 7	**IOW1WS**	**I/O Window 1 Wait State.** When Bit 7 is set to a one, 16-bit system accesses occur with one wait state (four System Clocks). A standard 16-bit I/O cycle completes in three System Clocks with IOCHRDY high.
Bit 6	**IOW1OW**	**I/O Window 1 Zero Wait State.** If bit 6 is set to a one, 8-bit system I/O accesses complete in three System Clocks, with ZEROWS# asserted to the system bus. However if bit 6 is set to a zero, the 8-bit system I/O access will complete in six System Clocks when the PC Card asserts no WAIT#, or more system clock cycles when WAIT# is asserted (cause IOCHRDY deasserted). WAIT# will override bit 6. This bit has no meaning for a 16-bit I/O access. 16-bit I/O accesses will always occur with either a three System Clock standard cycle when WAIT# is not active, or more cycles when WAIT# is active, or Bit 7 is set.
Bit 5	**IOW1IOCS**	**I/O Window 1 IOCS16# Source.** When bit 5 = '0' PCMCIA generates IOCS16# based on the value of the data size bit. However if bit 5 = '1', the PCMCIA generates IOCS16# based on IOS16# from the PC Card and the data size bit is ignored.
Bit 4		

Bit 3	**IOW0WS**	The above descriptions for bits 7-4 describing Window 1, are applicable to Window 0 as follows: **I/O Window 0 Wait State.**
Bit 2	**IOW0W**	**I/O Window 0 Zero Wait State.**
Bit 1	**IOW0IOC**	**I/O Window 0 IOCS16# Source.**
Bit 0	**IOW0DS**	**I/O Window 0 Data Size.**

18.11.2. I/O ADDRESS 0 START LOW BYTE REGISTER

This register contains the low order address bits used to determine the start address of I/O address window 0. This provides a minimum 1 byte window for I/O address window 0.

Add0_SLB Access = 0xCF8h/0xCFCh Regoffset = 0x08h

7	6	5	4	3	2	1	0
SA7-0							
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-0	**SA7-0**	**SA7-0.** These bits are to contain the low byte start address SA[7:0] for Window 0.

Programming notes:

A RESETDRV clears all bits in these registers, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = '1' in the slot A Power and RESETDRV Control Register.

18.11.3. I/O ADDRESS 0 START HIGH BYTE REGISTER

This register contains the high order address bits used to determine the start address of I/O address window 0.

Add0_SHB Access = 0xCF8h/0xCFCh Regoffset = 0x09h

7	6	5	4	3	2	1	0
SA15-8							
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-0	**SA15-8**	**SA15-8.** These bits are to contain the high byte start address SA15-8 for Window 0.

Programming notes:

A RESETDRV clears all bits in these registers, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = '1' in the slot Power and RESETDRV Control Register.

18.11.4. I/O ADDRESS 0 STOP LOW BYTE REGISTER

This register contains the low order address bits used to determine the stop address of I/O address window 0. This provides a minimum 1 byte window for I/O address window 0.

Note: No attempt should be made to overlay the I/O window over the top of the PCMCIA's registers. This will cause the PCMCIA access type to change.

Add0_SLB Access = 0xCF8h/0xCFCh Regoffset = 0x0Ah

7	6	5	4	3	2	1	0
SA7-0							
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-0	**SA7-0**	**SA7-0.** These bits are to contain the low byte low byte stop address SA7-0 for Window 0.

Programming notes:

A RESETDRV clears all bits in these registers, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = '1' in the slot Power and RESETDRV Control Register.

18.11.5. I/O ADDRESS 0 STOP HIGH BYTE REGISTER

This register contains the high order address bits used to determine the stop address of I/O address window 0.

Add0_SHB Access = 0xCF8h/0xCFCh Regoffset = 0x0Bh

7	6	5	4	3	2	1	0
SA15-8							
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-0	**SA15-8**	**SA15-8.** These bits are to contain the high byte stop address SA15-8 for Window 0.

Programming notes:

A RESETDRV clears all bits in these registers, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = '1' in the slot Power and RESETDRV Control Register.

18.11.6. I/O ADDRESS 1 START LOW BYTE REGISTER

This register contains the low order address bits used to determine the start address of I/O address window 1. This provides a minimum 1 byte window for I/O address window 1.

Add2_SLB Access = 0xCF8h/0xCFCh Regoffset = 0x0Ch

7	6	5	4	3	2	1	0
SA7-0							
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-0	**SA7-0**	**SA7-0.** These bits are to contain the low byte start address SA7-0 for Window 1.

Programming notes:

A RESETDRV clears all bits in these registers, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = '1' in the slot Power and RESETDRV Control Register.

18.11.7. I/O ADDRESS 1 START HIGH BYTE REGISTER

These registers contain the high order address bits used to determine the start address of I/O address window 1.

Add2_SHB Access = 0xCF8h/0xCFCh Regoffset = 0x0Dh

7	6	5	4	3	2	1	0
SA15-8							
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-0	**SA15-8**	**SA15-8.** These bits are to contain the high byte start address SA[15:8] for Window 1.

Programming notes:

A RESETDRV clears all bits in these registers, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = '1' in the slot Power and RESETDRV Control Register.

18.11.8. I/O ADDRESS 1 STOP LOW BYTE REGISTER

This register contains the low order address bits used to determine the stop address of I/O address window 1. This provides a minimum 1 byte window for I/O address window 1.

Note: No attempt should be made to overlay the I/O window over the top of the PCMCIA's registers. This will cause the PCMCIA access type to change.

Add2_StLB Access = 0xCF8h/0xCFCh Regoffset = 0x0Eh

7	6	5	4	3	2	1	0
SA7-0							
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-0	**SA7-0**	**SA7-0.** These bits are to contain the low byte stop address SA7-0 for Window 1.

Programming notes:

A RESETDRV clears all bits in these registers, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = '1' in the slot Power and RESETDRV Control Register.

18.11.9. I/O ADDRESS 1 STOP HIGH BYTE REGISTER

This register contains the high order address bits used to determine the stop address of I/O address window 1.

Add2_StHB Access = 0xCF8h/0xCFCh Regoffset = 0x0Fh

7	6	5	4	3	2	1	0
SA15-8							
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-0	**SA15-8**	**SA15-8.** These bits are to contain the high byte stop address SA15-8 for Window 1.

Programming notes:

A RESETDRV clears all bits in these registers, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = '1' in the slot Power and RESETDRV Control Register.

18.12. MEMORY REGISTERS

18.12.1. SYSTEM MEMORY ADDRESS 0 START LOW BYTE REGISTER

This register contains the low order address mapping bits used to determine the start address of the corresponding system memory address mapping window. This provides a minimum memory mapping window of 4K Bytes.

A memory PC Card is selected when the following conditions are satisfied:

1. The system memory address mapping window is enabled.
2. The (ISA) system memory address is greater than or equal to the system memory address mapping start register (high and low byte).
3. The (ISA) system memory address is less than or equal to the system memory address mapping stop register (high and low byte).

The system memory address mapping windows can all be configured by software, either independently or used together to perform mapping for special memory mapping requirements. Examples of such special requirements are LIM/EMS (Lotus-Intel-Microsoft/Extended Memory Specification), or XIP (Execute in Place).

Note: A memory window cannot be set up below the first 64K of address space.

Mem_Add0_SLB Access = 0xCF8h/0xCFCh Regoffset = 0x10h

7	6	5	4	3	2	1	0
SA7-0							
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-0	**SA7-0**	**SA7-0.** These bits are to contain the system memory low byte start address A[19:12] for Window 0.

Programming notes:

A RESETDRV clears all bits in these registers, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = '1' in the slot Power and RESETDRV Control Register.

18.12.2. SYSTEM MEMORY ADDRESS 0 START HIGH BYTE REGISTER

This register contains the high order address mapping bits used to determine the start address of the corresponding system memory address mapping window. Each system memory window has an associated data path size which is controlled by a bit in this register. Accesses to each system memory window have the potential to occur with zero additional wait states also controlled by a bit in this register.

Mem_Add0_SHB Access = 0xCF8h/0xCFCh Regoffset = 0x11h

7	6	5	4	3	2	1	0
DS	ZWS	SB		SMWS			
Default value after reset =							

Bit Number	Mnemonic	Description
Bit 7	**DS**	**Data Size.** When bit 7 = '0' an 8-bit memory data path to the PC Card is selected. If bit 7 = '1' then a 16-bit memory data path to the PC Card is selected.
Bit 6	**ZWS**	**Zero Wait State.** If bit 6 = '1', 8 or 16-bit system memory accesses complete in 3 or 2 System Clocks respectively, providing ZEROWS# is asserted to the system bus. However if bit 6 = '0', the 8 or 16-bit memory access will complete in 6 or 3 System Clocks respectively with IOCHRDY asserted high. In the event of IOCHRDY becoming de-asserted by internal wait state generation, or WAIT#, then setting bit 6 = '0' will cause 16-bit memory cycles to complete in more than 3 System Clocks. In the condition where IOCHRDY becomes de-asserted by WAIT# while bit 6 = '0', the 8-bit memory cycles will complete in more than 6 System Clocks. Only WAIT# can override bit 6. When bit 6 = '1' in this register, then the ZEROWS# output will be held high for accesses to an 8-bit system memory window with both A0 and SBHE# = '0'. **Note:** A logic low on IOCHRDY, either caused by an internal wait state generator or by WAIT# will force the ZEROWS# output high.
Bits 5-4	**SB**	**Scratch Bits.** Bits 5 and 4 are available to the designer for general purpose register storage and retrieval.
Bits 3-0	**SMWS**	**Syst. Mem. Window Start Address.** Bits 3 to 0 are high order address bits A[23:20] used to determine the start address of the corresponding system memory address mapping window.

Programming notes:

A RESETDRV clears all bits in these registers, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = '1' in the slot Power and RESETDRV Control Register.

18.12.3. SYSTEM MEMORY ADDRESS 0 STOP LOW BYTE REGISTER

This register contains the low order address bits used to determine the stop address of the corresponding system memory address mapping window. This provides a minimum memory mapping window of 4 K bytes.

Mem_Add0_StLB Access = 0xCF8h/0xCFCh Regoffset = 0x12h

7	6	5	4	3	2	1	0
SA19-12							
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-0	**SA19-12**	**SA19-12.** These bits are to contain the describe the required System Memory Window low byte stop address A19-12 for Window 0.

Programming notes:

A RESETDRV clears all bits in these registers, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = '1' in the slot Power and RESETDRV Control Register.

18.12.4. SYSTEM MEMORY ADDRESS 0 STOP HIGH BYTE REGISTER

This register contains the high order address mapping bits used to determine the stop address of the corresponding system memory address mapping window. All system memory windows have the ability to extend a 16-bit system bus cycle by inserting wait states. Two bits in each of these registers select the number of wait states for a 16-bit access to the system memory window.

Mem_Add0_StHB Access = 0xCF8h/0xCFCh Regoffset = 0x13h

7	6	5	4	3	2	1	0
WSS		Rsv		SA23-20			
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-6	**WSS**	**Wait State(s) Select.** Bits 7 and 6 determine the number of additional wait states for a 16-bit access to the system memory window. The internal wait state generator will not cause additional wait states to be inserted for an 8-bit system access, even if both bits = '1', because IOCHRDY will be pulled high by the PCMCIA before the system samples IOCHRDY. If the PC Card supports WAIT#, wait states will be generated by the PC Card asserting WAIT#. Bits 7 and 6 should be set to zero to disable the internal wait state generator. Table 18-13. shows the wait states that can be selected.
Bits 5-4	**Rsv**	**Reserved.** Bits 5 and 4 are reserved for internal use.
Bits 3-0	**SA23-20**	**SA23-SA20. Syst. Mem. Window Stop Address.** Bits 3 to 0 provide the high order address bits used to determine the stop address of the corresponding system memory address mapping window.

Table 18-13. Wait State Selection

Bit 7	Bit 6	Number of Additional Wait States	System Clocks per Access
0	0	Standard 16-Bit Cycle	3
0	1	1	4
1	0	2	5
1	1	3	6

Programming notes:

A RESETDRV clears all bits in these registers, unless RESETDRV is a result of a Host System resume (PWRGOOD = '1'), and the Disable Resume RESETDRV bit = '1' in the slot Power and RESETDRV Control Register.

18.12.5. CARD MEMORY OFFSET ADDRESS 0 LOW BYTE REGISTER

This register contains the low order address bits which are added to the system address bits A[19:12] to generate the memory address for the PC Card.

CM_Add0_LB Access = 0xCF8h/0xCFCh Regoffset = 0x14h

7	6	5	4	3	2	1	0
OA19-12							
Default value after reset =							

Bit Number	Mnemonic	Description
Bits 7-0	**OA19-12**	**OA19-OA12** These bits describe the required Card Memory Offset Address OA[19:12] for Window 0.

18.12.6. CARD MEMORY OFFSET ADDRESS 0 HIGH BYTE REGISTER

This register contains the high order address bits which are added to the system address bits A23-20 to generate the memory address for the PC Card. The software write protect of the PC Card memory for the corresponding system memory window is controlled by this register. Additionally the register also controls if the corresponding system memory window is mapped to attribute or common memory on the PC Card.

CM_Add0_HB Access = 0xCF8h/0xCFCh Regoffset = 0x15h

7	6	5	4	3	2	1	0
WP	RA	OA25-20					
Default value after reset =							

Bit Number	Mnemonic	Description
Bit 7	**WP**	**Write Protect.** If bit 7 = '1', write operations to the PC Card through the corresponding system memory window are inhibited. Conversely if bit 7 = '0' then these operations are enabled. It should be noted that the WP switch on the memory card by itself will not block the memory write cycle as it only sets the Memory Write Protect bit in the Interface Status Register.
Bit 6	**RA**	**Reg. Active.** When bit 6 = '1', accesses to the system memory window will result in attribute memory on the PC Card being accessed by asserting REG# to Low. However if bit 6 = '0', accesses to the system memory will result in common memory on the PC Card being accessed by the driving of REG# to High.
Bits 5-0	**OA25-20**	**OA25-20 Card Mem.Offset.** Bits 5 and 4 will be added to the system address bits A23-20 to generate the memory address for the PC Card.

18.12.7. SYSTEM MEMORY ADDRESSES 1-4 REGISTERS

The mapping registers for System Memory address 1 to 4 are identical to those described for address 0. Refer to Table 18-1. for the appropriate register addresses.

19. KEYBOARD / MOUSE CONTROLLER

19.1. INTRODUCTION

The Keyboard/Mouse Interface provides two PS/2 compatible serial interfaces with the industry standard 8042 compatible programming interface.

Both keyboard and mouse are supported, with the non-used function disabled via control of the KBCLK or MCLK outputs.

Full standard operation is provided, including input and output buffering, hardware parity check, time-out check, A20# control and complete scan code conversion for the 101 keys AT keyboard. For Keyboard with more than 101 keys, the scan code conversion must be deactivated and implemented in a seperate driver.

19.2. IO PINS

KBCLK, *Keyboard Clock line.* Keyboard data is latched by the controller on each negative clock edge produced on this pin. The keyboard can be disabled by pulling this pin low by software control.

KBDATA, *Keyboard Data Line.* 11 bits of data are shifted serially through this line when data is being transferred. Data is synchronised to KBCLK.

MCLK, *Mouse Clock line.* Mouse data is latched by the controller on each negative clock edge produced on this pin. The mouse can be disabled by pulling this pin low by software control.

MDATA, *Mouse Data Line.* 11 bits of data are shifted serially through this line when data is being transferred. Data is synchronised to MCLK.

The controller pins should be connected to the 14.318 MHz Series Cut Quartz Crystal oscillator to function properly.

19.3. FUNCTIONAL DESCRIPTION

19.3.1. BASIC OPERATION DURING KEYBOARD WRITE

Mouse interface is disabled by pulling the Mouse clock line low.

Keyboard clock line is pulled low for more than 60us so that if any keyboard read operation is in progress it is terminated.

When keyboard clock line is low the keyboard data line is pulled low.

After 60 us the keyboard clock line is pulled high and the controller expects that the keyboard starts the clock within 20ms. Failure to do so will result in a timeout .

When the keyboard starts the clock on each falling edge of the clock the controller shifts out the data. The data should be transmitted within 2 ms. Failure to do so will result in a timeout.

Once a write is over (i.e. 11 bits are transmitted), the controller waits for the keyboard to send an appropriate acknowledgement byte within 20 ms. Failing to do so will result in a timeout.

The received acknowledgement is placed in the output buffer, the OBF (Output Buffer Full) flag is set and the system is informed about the availability of the response through IRQ1.

During this time the mouse interface is kept disabled.

When OBF is set, both interfaces are disabled and will be enabled only when the system reads the output buffer i.e. OBF is reset.

19.3.2. BASIC OPERATION DURING KEYBOARD READ

Initially assume that both the interfaces are enabled (i.e. both Keyboard and Mouse Clock lines are high).

When Keyboard wants to send data, it pulls the Keyboard data line low and checks the status of Keyboard clock line. If the keyboard clock line is low, it implies that the keyboard interface is disabled and the keyboard makes its data line high. If the clock line is high, the keyboard starts driving the clock line.

Mouse interface is disabled by the controller by pulling the Mouse clock low.

With each rising edge of the clock, keyboard transmits one data bit on the data line. The data should be transmitted within 2 ms. Failure to do so will result in a timeout.

This data is latched by the controller on each negative clock edge.

After the transfer of all the 11 bits, the controller checks for the parity of the received byte. If a parity error is found, the Output Buffer register is loaded with ‘FFh’ and the OBF flag is set.

If there is no parity error and Keyboard is not locked, the data sent by the Keyboard (KSCAN code) is first converted into System Scan code. It is then loaded into the Output buffer and the OBF flag is set. However, if the Kscan code to System scan code conversion is disabled, the data is placed in the output buffer without any conversion.

The system is informed about the availability of the new byte through IRQ1.

When OBF is set, both the interfaces are disabled and will be enabled only when the system reads the output buffer i.e. OBF is reset.

19.3.3. BASIC OPERATION DURING MOUSE WRITE

Mouse Write is similar to Keyboard Write. In the description of Keyboard Write, interchanging Mouse Clock with Keyboard Clock, Mouse Data with Keyboard Data and replacing IRQ1 with IRQ12 will give the protocol for Mouse Write.

19.3.4. BASIC OPERATION DURING MOUSE READ

Mouse Read is similar to Keyboard Read. In the description of Keyboard Read, interchanging Mouse Clock with Keyboard Clock, Mouse Data with Keyboard Data and replacing IRQ1 with IRQ12 will give the protocol for Mouse Read. One difference for Mouse Read is that there is no conversion of data sent. The Mouse AUXBUF (Auxiliary Buffer Full) bit is also set by the controller.

The differences between Keyboard and Mouse are given in Table 19-1

Table 19-1. Differences Between Keyboard and Mouse

Address	Clock line	Data line	Int
Keyboard	KBCLK	KBDATA	IRQ1
Mouse	MCLK	MDATA	IRQ12

19.3.5. SPECIAL FEATURE

The keyboard and mouse controller that does not implement a feature that generally appears in other controllers. This cause a problem with software that assumes this feature is implemented. This feature is relative to commands sent to the keyboard or the mouse that are followed by data. The keyboard or mouse protocol is as follows: when a command is sent to the keyboard or the mouse, they will send an acknowledgement, then the data can be sent (there is also an acknowledge for the data) to the device.

In many controllers, the data can be sent just after sending the command, without waiting for the command acknowledge, the controller itself will send the data when it receives the acknowledge.

Because some keyboard or mouse software assumes this feature is implemented, it fails to send the appropriate commands with data to the keyboard or the mouse attached to the STPC controller.

In order to overcome this feature, please refer to the Appliation Note AN1332 that is located on the ST web site at www.st.com.

19.4. KEYBOARD/MOUSE CONTROLLER REGISTERS

The keyboard/mouse controller is accessed through IO Port 60h and 64h, as described in Table 19-2

I

Table 19-2. Keyboard / Mouse Controller Register Indexes

Address	Function	R/W#
60h	Input Buffer Data	Write
60h	Output Buffer	Read
64h	Input Buffer (Command)	Write
64h	Status Register	Read

19.4.1. INPUT BUFFER DATA REGISTER

This 8-bit register is written when a write operation is made to IO address 60h. The data written is the command data to be sent to the keyboard or to the mouse.

In_Buf Access = 0060h Regoffset =

7	6	5	4	3	2	1	0
Default value after reset =							

Before writing to this register, verify that it is empty, i.e. ensure that bit 1 (IBF) in the Status Register is set to 0.

19.4.2. OUTPUT BUFFER REGISTER

This 8-bit register is read when a read operation is made to IO address 60h. The data read is the keyboard scan code value for keyboard operation, the raw keyboard code if enabled, or the raw mouse data.

Out_Buf Access = 0060h Regoffset =

7	6	5	4	3	2	1	0
Default value after reset =							

19.4.3. COMMAND BYTE REGISTER

This 8-bit register is for keyboard controller command byte

Com_Byte Access = 0060h Regoffset =

7	6	5	4	3	2	1	0
Rsv	IBM_PC	M_CLK	KB_CLK	Rsv	Rsv	INT	OBF
Default value after reset =							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved**
Bit 6	**IBM_PC**	**IBM PC XT compatibility mode:** 0 = Compatibility disabled. This also disables the internal Scan Code table. 1 = Compatility enabled
Bit 5	**M_CLK**	**Disable Mouse (clock)** 0 = PS/2 Mouse disabled 1 = PS/2 Mouse enabled
Bit 4	**KB_CLK**	**Disable Keyboard (Clock)** 0 = KB disabled 1 = KB enabled
Bit 3	**Rsv**	**Reserved**
Bit 2	**Rsv**	**Reserved**
Bit 1	**INT**	**(PS/2) Enable Mouse Output Buffer Full interrupt (IRQ12)** 0 = Interrupt disabled 1 = Interrupt enabled
Bit 0	**OBF**	**Enable Keyboard Output Buffer Full interrupt (IRQ1)** 0 = Interrupt disabled 1 = Interrupt enabled

19.4.4. INPUT BUFFER (COMMAND) REGISTER

This 8-bit register is written when a write operation is made to IO address 64h.

Com_Reg Access = 0064h Regoffset =

7	6	5	4	3	2	1	0
Default value after reset =							

19.4.5. STATUS REGISTER

This 8-bit register is read when a read operation is made to IO address 64h.

Stat_Reg Access = 0064h Regoffset =

7	6	5	4	3	2	1	0
PARE	TIM	BUSY	Rsv	Data_Com	ERR	IBF	OBF
Default value after reset =							

Bit Number	Mnemonic	Description
Bit 7	**PARE**	**Parity Error Detected.** This bit is set if a parity error has been detected.
Bit 6	**TIM**	**Timeout Detected.** This bit is set if a time-out error has been detected.
Bit 5	**BUSY**	**Busy.** This bit indicates whether the keyboard/mouse controller is busy or available. This bit should be checked before any operation.
Bit 4	**Rsv**	**Reserved.**
Bit 3	**Data_Com**	**Input Register Status** 0 = Input written to the Input Register is data (Port 60) 1 = Input written to the Input Register is a command (Port 64h)
Bit 2	**Rsv**	**Reserved.**
Bit 1	**IBF**	**Input Buffer Full.** This bit indicates whether the Input Buffer is full, or whether it is available for receiving data or commands.
Bit 0	**OBF**	**Output Buffer Full.** This bit indicates whether the Output Buffer is full, or whether it is available for taking the data.

19.5. KEYBOARD/MOUSE CONTROLLER SUPPORTED COMMANDS

The Keyboard / Mouse Controller Commands supported by the STPC are outlined in Table 19-3.

Table 19-3. Supported KBM Controller Commands

Command	Command Description
20h	Read the current keyboard command byte. A 20h written to port 64h is followed by a read of port 60h.
60h	Write a new command byte to the keyboard controller. A command of 60h at port 64h is followed by a write to port 60h.
A7h -A8h	Disable mouse port and Enable Mouse port respectively. A disable mouse port command will set bit 5 of the command byte to 0 and pull down the Mouse clock line, preventing any data from being received or sent to the mouse. A enable mouse port command will set bit 5 of command byte to a 1 and the mouse clock line becomes active.
A9h	Test Mouse Port
ADh-AEh	Disable Keyboard and Enable Keyboard respectively.A disable keyboard port command will set bit 4 of the command byte to 0 and pull down the keyboard clock line, preventing any data from being received or sent to the keyboard. A enable keyboard port command will set bit 4 of command byte to a 1 and the keyboard clock line becomes active.
AAh	Keyboard Controller self test. In response to this command the controller will return 55h. Internally no tests will be carried out.
C0h	Read Input Port. Reads the controllers input port P1. A C0h command to port 64h followed by a read of 60h will return the contents of the input port. For bit definitions of the input port refer to Section
D0h -D1h	Read Output port and Write Output Port respectively. A read O/P port command followed by a read of port 60h will return the current status of the O/P port. A write o/p port command,D1h, followed by a write to port 60h with the appropriate byte will update the contents of the O/P port. The O/P Port is used to control the Gate A20 and generate a CPU Reset.
D4h	Write to Mouse port. Any command issued to the mouse should be preceded by this command to port 64h then any subsequent byte written to port 60h will be sent to mouse.
DDh-DFh	Disable A20 Address Line and Enable A20 Address Line respectively.
E0h	Read Test inputs
FEh	Generate System Reset. Issues hardware reset by setting the system reset line low for approximately 6 microseconds.

20. LOCAL BUS INTERFACE

20.1. INTRODUCTION

The Local Bus interface of the STPC provides a low latency bus to external peripheral. The Local Bus may operate the 25-bit address and 16-bit data bus.

The Local Bus interface supports up to two memory banks and eight I/O devices. It can support up to 32 MBytes of memory in each memory bank and from 1 Byte to 1 KByte of I/O space for each of the I/O devices. All the chip select timings are individually programmable. This interface can be accessed only by the CPU.

The first bank of the memory is intended to be used as the boot device.

The starting address for each I/O chip select is programmable at the 1 Byte boundary. The access starting range for each of the chip selects is also programmable. The size varies from 1 Byte to 1 KByte.

Note: The base address of any local bus slot (I/O or memory) has to be an integral multiple of the size of the slot. For example, a 32 MByte flash must have a 32 MByte aligned base address.

20.1.1. FEATURES

- Support of 8/16/32-bit cycles for both 8/16-bit I/O or memory devices
- Two banks of 32 MByte (max.) each, one as a boot device
- Memory Banks size and programmable base address
- Programmable timing with host clock granularity for Bank and I/O accesses
- Up to eight I/O devices supported with programmable start address & size
- I/O device timing (setup & recovery time) programmable
- Interrupt support

20.2. MEMORY BANK SWITCHING

The Local Bus Interface caters for two memory banks of up to 32 MBytes (max), designated Bank 0 (upper) and Bank 1 (lower). Since 32 MByte Flash memory devices are not commercially available, to support the total 64 MByte capacity, four Flash Chip Select pins, listed below, are provided to allow the use of up to four 16 Mb Flash devices.

FCS0#: Chip Select for Bank 0

FCS1#:Chip Select for Bank 1

FCS_0H: Chip select for High (upper) region of Bank 0

FCS_0L:Chip select for Low (lower) region of Bank 0

FCS_1H:Chip select for High (upper) region of Bank 1

FCS_1L:Chip select for Low (lower) region of Bank 1

The base addresses for the two Banks are specified in registers MEMAREG0 (Section 20.5.6.) and MEMAREG1 (Section 20.5.7.). The size of memory allocated to each of the two Banks is specified in the MEMMASK register (Section 20.5.8.). Note that the minimum size that can be allocated to a Bank is 1 MByte.

20.3. FLASH DEVICE IMPLEMENTATION

The Local Bus Controller is capable of managing four 16MByte flash devices in 8 or 16bit wide configuration allowing great flexibility of implimentation.

The choice of configuration depends on the boot method required;

20.3.1. STANDARD BIOS BOOT OR BOOT LOADER IN REAL MODE

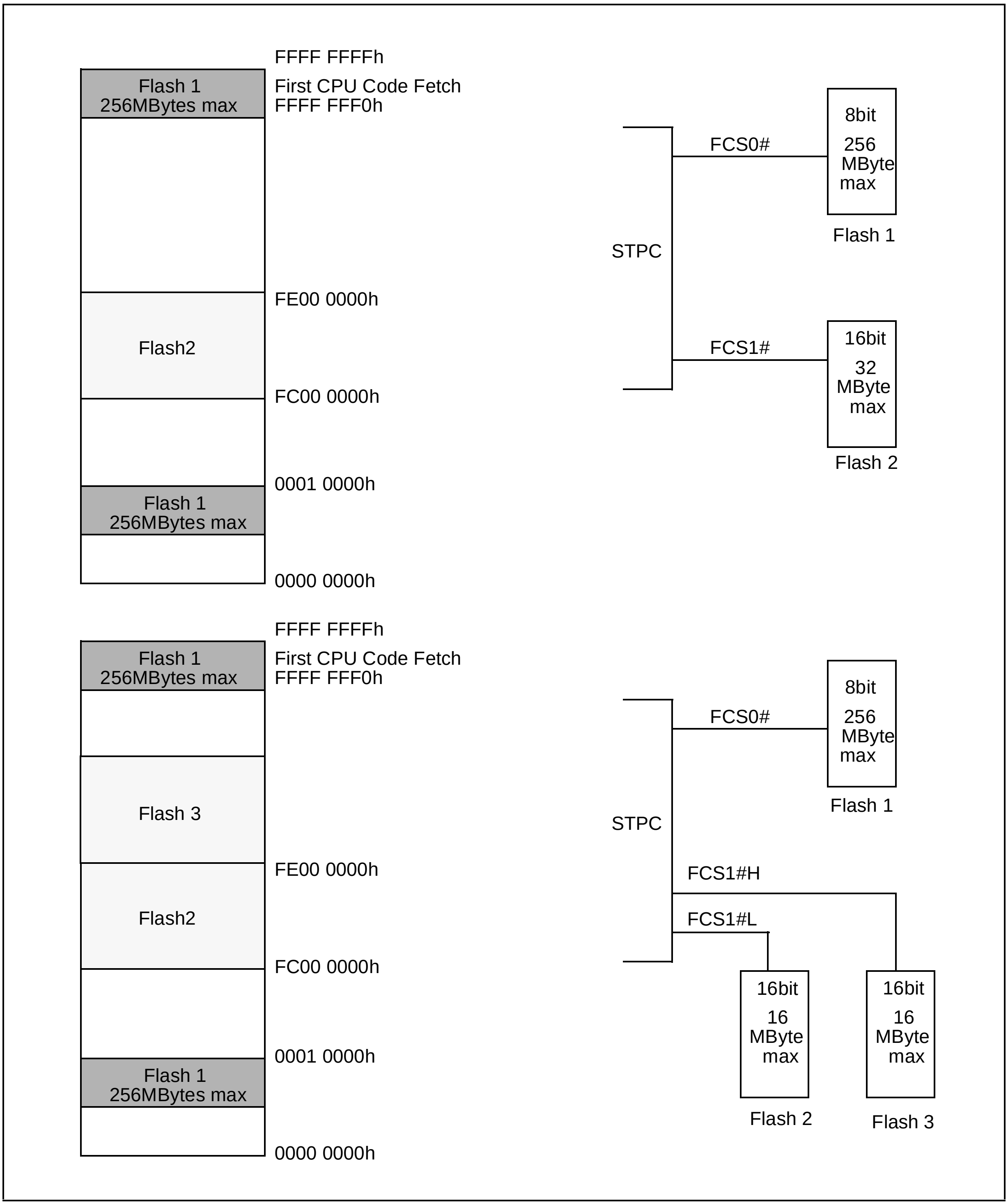
In order to execute boot code below the 1st MByte of memory, the flash device must imperatively be connected to FCS0#, 8 bit wide and have a maximum capacity of 256 KBytes. In this case, bank 1 is used in the same as an ISA bus implementation.

Bank 1 may be used to store data or an OS which is loaded by the code stored in bank 0.

The limitation of this implimentation is that only 32MBytes are available for Data and OS storage and booting in 8bit configuration is available.

This is illustrated in Figure 20-1.

Figure 20-1. Standars BIOS Boot Illustration



20.3.2. BOOTLOADER THAT IS EXECUTED ABOVE THE FIRST MBYTE

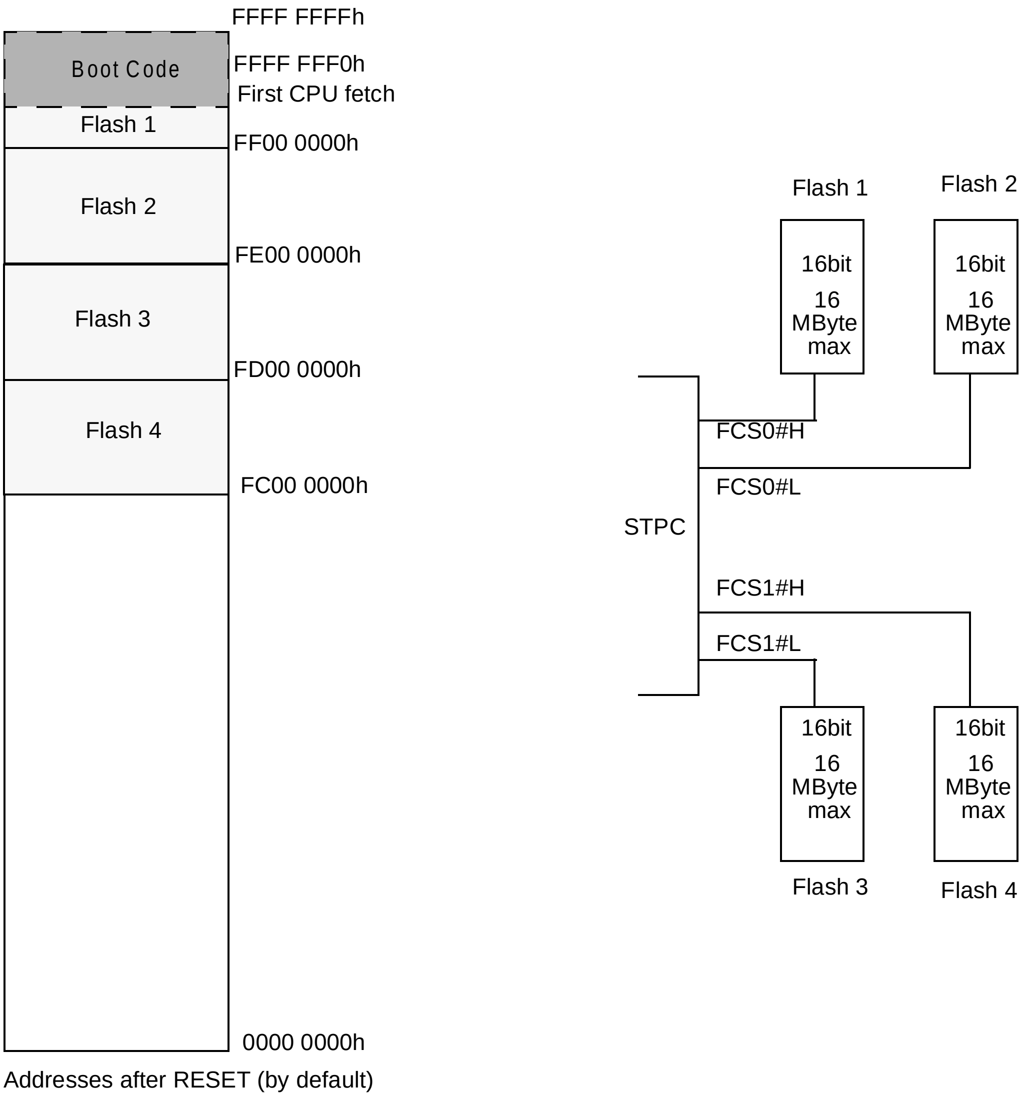
In order to use high capacity 16bit flash devices on Bank 0, the boot code must be executed above the 1MByte. The first CPU fetch is carried out at FFFF FFF0h (4GByte - 16Byte).

The CPU is only able to see flash devices that are located above 1MBytes of memory. Therefore, the boot code needs to initialise the memory and copy itself below the first MByte before it is able to use a stack, make function calls or to modify the Code Segment Register CS (which is equivalent of readig the next CPU instructions under the first MByte).

To simplify the developer's task, a primary bootloader has been developped by STMicroelectronics. This bootloader initialises the STPC, detects and configures the memory before being copied below the first MByte, and hands over to next boot programme (Secondary loader).

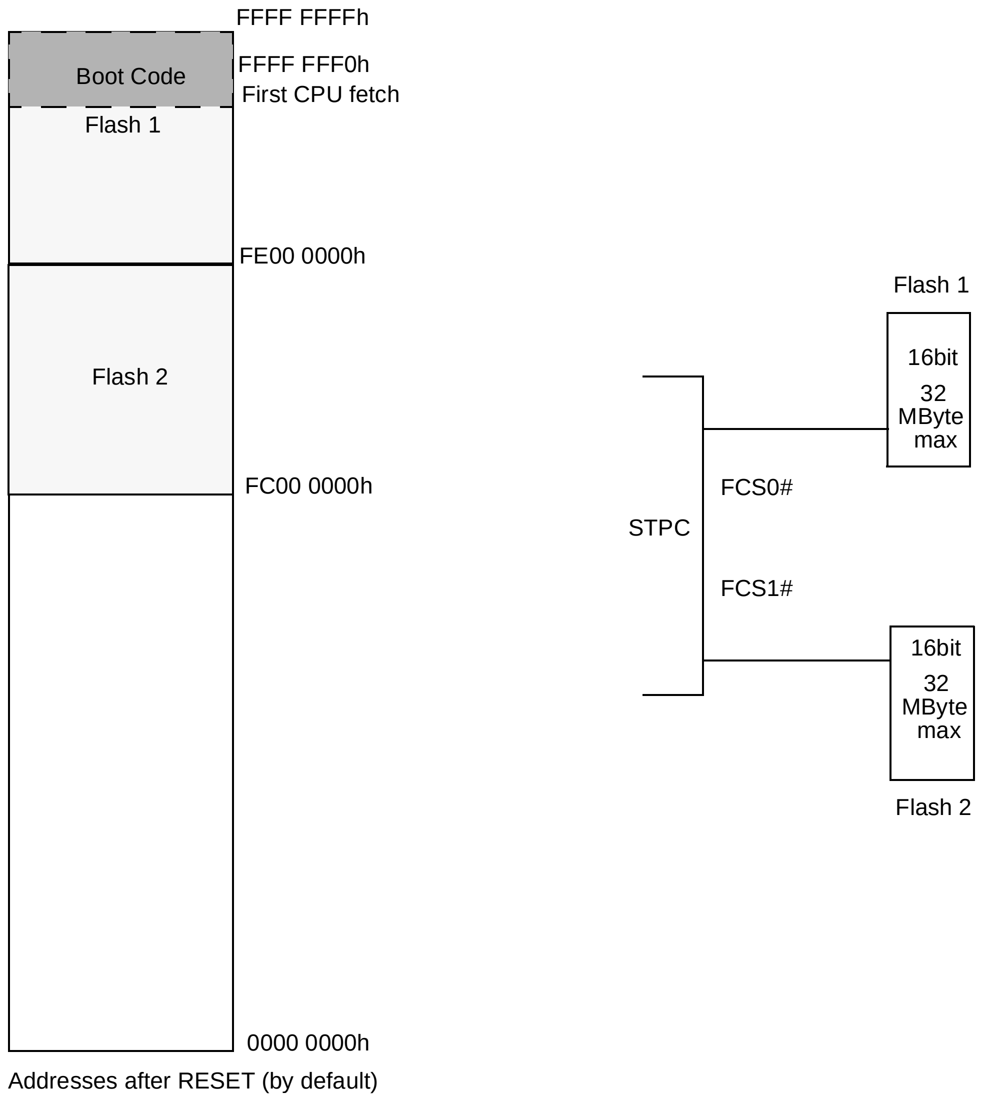
This is illustrated in Figure 20-2 and Figure 20-3.

Figure 20-2. Four flash device implimentation



Note; In this case the Bank size can't be modified and
the base address can only be changed at a Modulo of 32 MBytes

Figure 20-3. Two flash device implementation



Note; In this case the Bank size can't be modified and
the base address can only be changed at a Modulo of 32 MBytes

20.4. CONFIGURATION REGISTERS

The Local Bus configuration registers can be categorised into five groups:

1) Base Index Register
2) Address Decode Registers
3) Timing Registers
4) Control Register
5) Device Width Register

Table 20-1. 16-bit Address Decode Registers for I/O and MEM

Offset	**Register Name**	**Register Description**	**Default Values**
Variable, see Section 20.5.	BASE_INDEX_REG[15:0]	*Base_Index_Reg* for Configuration registers	Base
0x0000h	IOAREG0[15:0] Base IO 0	Base Address of I/O Slots 0	0xFFFFh
0x0002h	IOAREG1[15:0] Base IO 1	Base Address of I/O Slots 1	0xFFFFh
0x0004h	IOAREG2[15:0] Base IO 2	Base Address of I/O Slots 2	0xFFFFh
0x0006h	IOAREG3[15:0] Base IO 3	Base Address of I/O Slots 3	0xFFFFh
0x0008h	IOAREG4[15:0] Base IO 4	Base Address of I/O Slots 4	0xFFFFh
0x000Ah	IOAREG5[15:0] Base IO 5	Base Address of I/O Slots 5	0xFFFFh
0x000Ch	IOAREG6[15:0] Base IO 6	Base Address of I/O Slots 6	0xFFFFh
0x000Eh	IOAREG7[15:0] Base IO 7	Base Address of I/O Slots 7	0xFFFFh
0x0010h	IOMREG0[15:0] Mask IO0	Mask Size of I/O Slots 0	0xFFFFh
0x0012h	IOMREG1[15:0] Mask IO1	Mask Size of I/O Slots 1	0xFFFFh
0x0014h	IOMREG2[15:0] Mask IO2	Mask Size of I/O Slots 2	0xFFFFh
0x0016h	IOMREG3[15:0] Mask IO3	Mask Size of I/O Slots 3	0xFFFFh
0x0018h	IOMREG4[15:0] Mask IO4	Mask Size of I/O Slots 4	0xFFFFh
0x001Ah	IOMREG5[15:0] Mask IO5	Mask Size of I/O Slots 5	0xFFFFh
0x001Ch	IOMREG6[15:0] Mask IO6	Mask Size of I/O Slots 6	0xFFFFh
0x001Eh	IOMREG7[15:0] Mask IO7	Mask Size of I/O Slots 7	0xFFFFh

Table 20-1. 16-bit Address Decode Registers for I/O and MEM

Offset	Register Name	Register Description	Default Values
0x0020h	TIMEMEM0[15:0]	Timing Template for accessing bank 0	0xA87Dh
0x0022h	TIMEMEM1[15:0]	Timing Template for accessing bank 1	0xA87Dh
0x0024h	TIMEIO0[15:0]	Timing Template for accessing I/O 0	0x0000h
0x0026h	TIMEIO1[15:0]	Timing Template for accessing I/O 1	0x0000h
0x0028h	TIMEIO2[15:0]	Timing Template for accessing I/O 2	0x0000h
0x002Ah	TIMEIO3[15:0]	Timing Template for accessing I/O 3	0x0000h
0x002Ch	TIMEIO4[15:0]	Timing Template for accessing I/O 4	0x0000h
0x002Eh	TIMEIO5[15:0]	Timing Template for accessing I/O 5	0x0000h
0x0030h	TIMEIO6[15:0]	Timing Template for accessing I/O 6	0x0000h
0x0032h	TIMEIO7[15:0]	Timing Template for accessing I/O 7	0x0000h
0x0034h	CONTROL[15:0]	16-bit Control Register	0x0001h
0x0036h	DEVICE_WIDTH[15:0]	I/O or MEM width as 8-bit or 16-bit	0x0300h
0x0038h	MEMAREG0[15:0]	Base address [31:20]of Memory bank0	0x0FE0h
0x003Ah	MEMAREG1[15:0]	Base address [31:20]of Memory bank1	0x0FC0h
0x003Ch	MEMMASK[15:0]	Address range of Memory bank0 & 1	0x003Fh

20.5. LOCAL BUS BASE INDEX REGISTER

The Atlas local bus base address is variable. Access can be calculated using using the Pseudo code given below. The following assumes that the Local Bus (LB) is device number 6, and that the LB I/O index is 28h with the data at 2Ch.

20.5.1. INITIALISATION

Select the Local Bus device:

```
IOWRITE8 (22h, 10h)
IOWRITE8 (23h, 06h)
IOWRITE8 (22h, 11h)
IOWRITE8 (23h, 00h)
```

Set the Local Bus base address and enable access:

```
IOWRITE8 (22h, 12h)
IOWRITE8 (23h, 28h)
IOWRITE8(23h,(HOST_BASE &FFh) | 03h);
IOWRITE8 (22h, 13h)
IOWRITE8 (23h, 00h)
IOWRITE8(23h,HOST_BASE >>8);
```

20.5.2. WRITE ACCESS

```
IOWRITE16 (HOST_BASE, register_index)
IOWRITE16 (HOST_BASE + 4, data)
```

20.5.3. READ ACCESS

```
IOWRITE16 (HOST_BASE, register_index)
Data = IOREAD16 (HOST_BASE + 4)
```

Note that further information is available in Section 6.11.

20.5.4. I/O SLOT BASE ADDRESS REGISTERS IOAREG0 TO IOAREG7

IOAREG# Access = see Section 20.5. Regoffset: see Table 20-1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SA															
Default value after reset = FFFFh															

Bit Number	Mnemonic	Description
Bits 15-0	**SA**	Starting Address aligned to a Byte.

20.5.5. I/O SLOT MASK REGISTERS

The eight address mask registers IOMREG0 to IOMREG7 define the size of each I/O slot. The 10-bit address mask will mask the part of the address that is not to be used during the address decoding process. The 10-bit mask will filter bit 9:0 of the starting address specified in the corresponding IOAREG slot base register.

IOMREG# Access = see Section 20.5.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv						AMIO									
Default value after reset = FFFFh															

Bit Number	Mnemonic	Description
Bits 15-10	**Rsv**	**Reserved**
Bits 9-0	**AMIO**	Bits used for address masking.

Register	Regoffset
IOMREG0	10h
IOMREG1	12h
IOMREG2	14h
IOMREG3	16h
IOMREG4	18h
IOMREG5	1Ah
IOMREG6	1Ch
IOMREG7	1Eh

20.5.6. MEMORY BASE ADDRESS REGISTER 0

The Base Address for Bank0 is specified by base address register MEMAREG0. The minimum size that a bank can have is 1Mb.

MEMAREG0 Access = see Section 20.5. Regoffset = 38h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv															
Default value after reset = 0FE0h															

Bit Number	Mnemonic	Description
Bits 15-12	**Rsv**	**Reserved**
Bits 11-0		Bits 31:20 of memory base for Bank0

Bank0 has a Base Address register mapped at 0xFE000000h after reset, and having a size of 32Mb (max).

20.5.7. MEMORY BASE ADDRESS REGISTER 1

The Base Address for Bank1 is specified by base address register MEMAREG1. The minimum size that a bank can have is 1Mb.

MEMAREG1 Access = see Section 20.5. Regoffset = 3Ah

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv															
Default value after reset = 0FC0h															

Bit Number	Mnemonic	Description
Bits 15-12	**Rsv**	**Reserved**
Bits 11-0		Bits 31:20 of memory base for Bank1

Note: The Base for Bank1 should be anything above the physical DRAM address and up to anything below (0xFE000000 - size of Bank1).

For example, for a Base Address of 0xF0000000h for bank1, MEMAREG1 = 0F00h.

20.5.8. MEMORY MASK REGISTER

The MEMMASK register defines the memory sizes for Bank0 and Bank1.

MEMMASK Access = see Section 20.5. Regoffset = 3Ch

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv								Rsv							
Default value after reset = 003Fh															

Bit Number	Mnemonic	Description
Bits 15-14	**Rsv**	**Reserved**
Bits 13-9		**Mask of Memory Bank1 in Mb**
Bit 8		**Enable/Disable Memory Bank1** 0 = disable, 1 = enable
Bits 7-6	**Rsv**	**Reserved**
Bits 5-1		**Mask of Memory Bank0 in Mb**
Bit 0		**Enable/Disable Memory Bank0** 0 = disable, 1 = enable

Note: It is recommended that the Mask should be specified in powers of two -1, as in the following Table, so that the memory size could be 1Mb, 2 Mb, 4 Mb, 8 Mb, 16 Mb or 32 Mb.

Coding Bits 13:9 (Bank1) & 5:1 (Bank0)					**Mask Size**
0	0	0	0	0	1 Mb
0	0	0	0	1	2 Mb
0	0	0	1	1	4 Mb
0	0	1	1	1	8 Mb
0	1	1	1	1	16 Mb
1	1	1	1	1	32 Mb
Any other value not mentioned in the above is forbidden					

The default value (001Fh) represents 32 Mb max. and bank0 enabled, bank1 disabled.

20.6. LOCAL BUS TIMING REGISTERS

20.6.1. TIMING MEMORY TEMPLATE REGISTER 0

TIMEMEM0 defines the timing template for accessing Flash memory Bank0. The timing is programmed with reference to the host clock period as a time unit.

TIMEMEM0 Access = see Section 20.5. Regoffset = 20h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSH	UARS	CHT			CAT								CST		
Default value after reset = A87Dh															

Bit Number	Mnemonic	Description
Bit 15	**DSH**	**Discard the Setup and Hold parameters for read operations.** 0 = Do not discard, 1 = discard
Bit 14	**UARS**	**Use asynchronous ready signal for command termination.** The asynchronous ready enable bit sets the Local Bus logic to wait for an external ready signal before closing the current cycle. 0 = Do not use, 1 = Use
Bits 13-11	**CHT**	**Command hold time,** and is determined as follows: Command hold time = (4+Vh) x T where Vh = Register value for the Hold time T = HCLK period
Bits 10-3	**CAT**	**Command Active time,** and is determined as follows: Command active time = (2+Va) x T where Va = Register value for the Active time T = HCLK period.
Bits 2-0	**CST**	**Command Setup time,** and is determined as follows: Command setup time = (4+Vs) x T where Vs = Register value for the Setup time T = HCLK period.

20.6.2. TIMING MEMORY TEMPLATE REGISTER 1

TIMEMEM1 defines the timing template for accessing Flash memory Bank1. The timing is programmed with reference to the host clock period as a time unit.

TIMEMEM1 Access = see Section 20.5. Regoffset = 22h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSH	UARS	CHT			CAT								CST		
Default value after reset = A87Dh															

Bit Number	Mnemonic	Description
Bit 15	**DSH**	**Discard the Setup and Hold parameters for read operations.** 0 = Do not discard, 1 = discard
Bit 14	**UARS**	**Use asynchronous ready signal for command termination.** The asynchronous ready enable bit sets the Local Bus logic to wait for an external ready signal before closing the current cycle. 0 = Do not use, 1 = Use
Bits 13-11	**CHT**	**Command hold time,** and is determined as follows: Command hold time = (4+Vh) x T where Vh = Register value for the Hold time T = HCLK period
Bits 10-3	**CAT**	**Command Active time,** and is determined as follows: Command active time = (2+Va) x T where Va = Register value for the Active time T = HCLK period.
Bits 2-0	**CST**	**Command Setup time,** and is determined as follows: Command setup time = (4+Vs) x T where Vs = Register value for the Setup time T = HCLK period.

20.6.3. I/O TIMING TEMPLATE REGISTER 0

TIMEIO0 defines the timing template for accessing device in I/O Slot 0. The timing is programmed with reference to the host clock period as a time unit.

TIMEIO0 Access = see Section 20.5. Regoffset = 24h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSH	UARS	CHT			CAT								CST		
Default value after reset = 0000h															

Bit Number	Mnemonic	Description
Bit 15	**DSH**	**Discard the Setup and Hold parameters for read operations.** 0 = Do not discard, 1 = discard
Bit 14	**UARS**	**Use Asynchronous Ready Signal for command termination.** The asynchronous ready enable bit sets the Local Bus logic to wait for an external ready signal before closing the current cycle. 0 = Do not use, 1 = Use
Bits 13-11	**CHT**	**Command Hold Time,** and is determined as follows: Command hold time = (4+Vh) x T where Vh = Register value for the Hold time T = HCLK period
Bits 10-3	**CAT**	**Command Active Time,** and is determined as follows: Command active time = (2+Va) x T where Va = Register value for the Active time T = HCLK period.
Bits 2-0	**CST**	**Command Setup Time,** and is determined as follows: Command setup time = (4+Vs) x T where Vs = Register value for the Setup time T = HCLK period.

20.6.4. I/O TIMING TEMPLATE REGISTER 1

TIMEIO1 defines the timing template for accessing device in I/O Slot 1. The timing is programmed with reference to the host clock period as a time unit.

TIMEIO1 Access = see Section 20.5. Regoffset = 26h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSH	UARS	CHT			CAT								CST		
Default value after reset = 0000h															

Bit Number	Mnemonic	Description
Bit 15	**DSH**	**Discard the Setup and Hold parameters for read operations.** 0 = Do not discard, 1 = discard
Bit 14	**UARS**	**Use Asynchronous Ready Signal for command termination.** The asynchronous ready enable bit sets the Local Bus logic to wait for an external ready signal before closing the current cycle. 0 = Do not use, 1 = Use
Bits 13-11	**CHT**	**Command Hold Time,** and is determined as follows: Command hold time = (4+Vh) x T where Vh = Register value for the Hold time T = HCLK period
Bits 10-3	**CAT**	**Command Active Time,** and is determined as follows: Command active time = (2+Va) x T where Va = Register value for the Active time T = HCLK period.
Bits 2-0	**CST**	**Command Setup Time,** and is determined as follows: Command setup time = (4+Vs) x T where Vs = Register value for the Setup time T = HCLK period.

20.6.5. I/O TIMING TEMPLATE REGISTER 2

TIMEIO2 defines the timing template for accessing device in I/O Slot 2. The timing is programmed with reference to the host clock period as a time unit.

TIMEIO2 Access = see Section 20.5. Regoffset = 28h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSH	UARS	CHT			CAT								CST		
Default value after reset = 0000h															

Bit Number	Mnemonic	Description
Bit 15	**DSH**	**Discard the Setup and Hold parameters for read operations.** 0 = Do not discard, 1 = discard
Bit 14	**UARS**	**Use Asynchronous Ready SIgnal for command termination.** The asynchronous ready enable bit sets the Local Bus logic to wait for an external ready signal before closing the current cycle. 0 = Do not use, 1 = Use
Bits 13-11	**CHT**	**Command Hold Time,** and is determined as follows: Command hold time = (4+Vh) x T where Vh = Register value for the Hold time T = HCLK period
Bits 10-3	**CAT**	**Command Active Time,** and is determined as follows: Command active time = (2+Va) x T where Va = Register value for the Active time T = HCLK period.
Bits 2-0	**CST**	**Command Setup Time,** and is determined as follows: Command setup time = (4+Vs) x T where Vs = Register value for the Setup time T = HCLK period.

20.6.6. I/O TIMING TEMPLATE REGISTER 3

TIMEIO3 defines the timing template for accessing device in I/O Slot 3. The timing is programmed with reference to the host clock period as a time unit.

TIMEIO3 Access = see Section 20.5. Regoffset = 2Ah

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSH	UARS	CHT			CAT								CST		
Default value after reset = 0000h															

Bit Number	Mnemonic	Description
Bit 15	**DSH**	**Discard the Setup and Hold parameters for read operations.** 0 = Do not discard, 1 = discard
Bit 14	**UARS**	**Use Asynchronous Ready Signal for command termination.** The asynchronous ready enable bit sets the Local Bus logic to wait for an external ready signal before closing the current cycle. 0 = Do not use, 1 = Use
Bits 13-11	**CHT**	**Command Hold Time,** and is determined as follows: Command hold time = (4+Vh) x T where Vh = Register value for the Hold time T = HCLK period
Bits 10-3	**CAT**	**Command Active Time,** and is determined as follows: Command active time = (2+Va) x T where Va = Register value for the Active time T = HCLK period.
Bits 2-0	**CST**	**Command Setup Time,** and is determined as follows: Command setup time = (4+Vs) x T where Vs = Register value for the Setup time T = HCLK period.

20.6.7. I/O TIMING TEMPLATE REGISTER 4

TIMEIO4 defines the timing template for accessing device in I/O Slot 4. The timing is programmed with reference to the host clock period as a time unit.

TIMEIO4 Access = see Section 20.5. Regoffset = 2Ch

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSH	UARS	CHT			CAT								CST		
Default value after reset = 0000h															

Bit Number	Mnemonic	Description
Bit 15	**DSH**	**Discard the Setup and Hold parameters for read operations.** 0 = Do not discard, 1 = discard
Bit 14	**UARS**	**Use Asynchronous Ready Signal for command termination.** The asynchronous ready enable bit sets the Local Bus logic to wait for an external ready signal before closing the current cycle. 0 = Do not use, 1 = Use
Bits 13-11	**CHT**	**Command Hold Time,** and is determined as follows: Command hold time = (4+Vh) x T where Vh = Register value for the Hold time T = HCLK period
Bits 10-3	**CAT**	**Command Active Time,** and is determined as follows: Command active time = (2+Va) x T where Va = Register value for the Active time T = HCLK period.
Bits 2-0	**CST**	**Command Setup Time,** and is determined as follows: Command setup time = (4+Vs) x T where Vs = Register value for the Setup time T = HCLK period.

20.6.8. I/O TIMING TEMPLATE REGISTER 5

TIMEIO5 defines the timing template for accessing device in I/O Slot 5. The timing is programmed with reference to the host clock period as a time unit.

TIMEIO5 Access = see Section 20.5. Regoffset = 2Eh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSH	UARS	CHT			CAT								CST		
Default value after reset = 0000h															

Bit Number	Mnemonic	Description
Bit 15	**DSH**	**Discard the Setup and Hold parameters for read operations.** 0 = Do not discard, 1 = discard
Bit 14	**UARS**	**Use Asynchronous Ready Signal for command termination.** The asynchronous ready enable bit sets the Local Bus logic to wait for an external ready signal before closing the current cycle. 0 = Do not use, 1 = Use
Bits 13-11	**CHT**	**Command Hold Time,** and is determined as follows: Command hold time = (4+Vh) x T where Vh = Register value for the Hold time T = HCLK period
Bits 10-3	**CAT**	**Command Active Time,** and is determined as follows: Command active time = (2+Va) x T where Va = Register value for the Active time T = HCLK period.
Bits 2-0	**CST**	**Command Setup Time,** and is determined as follows: Command setup time = (4+Vs) x T where Vs = Register value for the Setup time T = HCLK period.

20.6.9. I/O TIMING TEMPLATE REGISTER 6

TIMEIO6 defines the timing template for accessing device in I/O Slot 6. The timing is programmed with reference to the host clock period as a time unit.

TIMEIO6 Access = see Section 20.5. Regoffset = 30h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSH	UARS	CHT			CAT								CST		
Default value after reset = 0000h															

Bit Number	Mnemonic	Description
Bit 15	**DSH**	**Discard the Setup and Hold parameters for read operations.** 0 = Do not discard, 1 = discard
Bit 14	**UARS**	**Use Asynchronous Ready Signal for command termination.** The asynchronous ready enable bit sets the Local Bus logic to wait for an external ready signal before closing the current cycle. 0 = Do not use, 1 = Use
Bits 13-11	**CHT**	**Command Hold Time,** and is determined as follows: Command hold time = (4+Vh) x T where Vh = Register value for the Hold time T = HCLK period
Bits 10-3	**CAT**	**Command Active Time,** and is determined as follows: Command active time = (2+Va) x T where Va = Register value for the Active time T = HCLK period.
Bits 2-0	**CST**	**Command Setup Time,** and is determined as follows: Command setup time = (4+Vs) x T where Vs = Register value for the Setup time T = HCLK period.

20.6.10. I/O TIMING TEMPLATE REGISTER 7

TIMEIO7 defines the timing template for accessing device in I/O Slot 7. The timing is programmed with reference to the host clock period as a time unit.

TIMEIO7 Access = see Section 20.5. Regoffset = 32h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSH	UARS	CHT			CAT								CST		
Default value after reset = 0000h															

Bit Number	Mnemonic	Description
Bit 15	**DSH**	**Discard the Setup and Hold parameters for read operations.** 0 = Do not discard, 1 = discard
Bit 14	**UARS**	**Use Asynchronous Ready Signal for command termination.** The asynchronous ready enable bit sets the Local Bus logic to wait for an external ready signal before closing the current cycle. 0 = Do not use, 1 = Use
Bits 13-11	**CHT**	**Command Hold Time,** and is determined as follows: Command hold time = (4+Vh) x T where Vh = Register value for the Hold time T = HCLK period
Bits 10-3	**CAT**	**Command Active Time,** and is determined as follows: Command active time = (2+Va) x T where Va = Register value for the Active time T = HCLK period.
Bits 2-0	**CST**	**Command Setup Time,** and is determined as follows: Command setup time = (4+Vs) x T where Vs = Register value for the Setup time T = HCLK period.

20.7. LOCAL BUS CONTROL REGISTER

This 16-bit register controls the basic functionality of the local bus interface.

CONTROL Access = see Section 20.5. Regoffset = 34h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv															
Default value after reset = 0001h															

Bit Number	Mnemonic	Description
Bits 15-11	**Rsv**	**Reserved**
Bit 10	**ISE**	**Interrupt Support Enable** 0 = Disable, 1 = Enable
Bit 9	**EMCB0**	**Enable Mapping to segment C of Bank0**[2] 0 = Disable, 1 = Enable
Bit 8	**EMDB0**	**Enable Mapping to segment D of Bank0**[2] 0 = Disable, 1 = Enable
Bit 7	**EMEB0**	**Enable Mapping to segment E of Bank0**[2] 0 = Disable, 1 = Enable
Bit 6		**Splitting for 8-bit device** 0 = use bs 8 signal generation 1 = use internal state machine logic
Bit 5		**Splitting for 16-bit device** 0 = use bs 16 signal generation 1 = use internal state machine logic
Bit 4	**CEB1**	**Cache Enable for Bank1**, 0 = Disable, 1 = Enable
Bit 3	**CEB0**	**Cache Enable for Bank0**, 0 = Disable, 1 = Enable
Bit 2	**WEB1**	**Write Enable for Bank1**, 0 = Disable, 1 = Enable
Bit 1	**WEB0**	**Write Enable for Bank0**, 0 = Disable, 1 = Enable
Bit 0	**RMBAE**	**Real Mode Boot Access Enable** 0 = Disable, 1 = Enable[1]

Notes:

1) Generation of Chip select for Bank0 at 0x000Fh:XXXX address.

2) Enable mapping to C,D,E segments means enabling Bank0 for address range 0x000Ch:XXXX, 0x000Dh:XXXX and 0x000Eh:XXXX respectively.

20.8. LOCAL BUS DEVICE WIDTH REGISTER

This is a 16-bit register whose individual bits are used to tell the Local Bus if an 8 or 16-bit peripheral is attached to one of the eight I/O and two MEM slots. Only ten LSB bits of the register are used.

WIDTH Access = see Section 20.5. Regoffset = 36h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv						DWB1	DWB0	IOW7	IOW6	IOW5	IOW4	IOW3	IOW2	IOW1	IOW0
Default value after reset = 0300h															

Bit Number	Mnemonic	Description
Bits 15-10	**Rsv**	**Reserved**
Bit 9	**DWB1**	**Data Width of Bank1** 0 = 8-bit, 1 = 16-bit
Bit 8	**DWB0**	**Data Width of Bank0**[1] 0 = 8-bit, 1 = 16-bit
Bit 7	**IOW7**	**I/O 7 Data Width** 0 = 8-bit, 1 = 16-bit
Bit 6	**IOW6**	**I/O 6 Data Width** 0 = 8-bit, 1 = 16-bit
Bit 5	**IOW5**	**I/O 5 Data Width** 0 = 8-bit, 1 = 16-bit
Bit 4	**IOW4**	**I/O 4 Data Width** 0 = 8-bit, 1 = 16-bit
Bit 3	**IOW3**	**I/O 3 Data Width** 0 = 8-bit, 1 = 16-bit
Bit 2	**IOW2**	**I/O 2 Data Width** 0 = 8-bit, 1 = 16-bit
Bit 1	**IOW1**	**I/O 1 Data Width** 0 = 8-bit, 1 = 16-bit
Bit 0	**IOW0**	**I/O 0 Data Width** 0 = 8-bit, 1 = 16-bit

Note:This is also controlled by a strap option input to Local Bus:

Strap bit = 1: The data width of Bank0 is 16-bit

Strap bit = 0: The data width of Bank0 is 8-bit.

21. GPIO INTERFACE

21.1. INTRODUCTION

The GPIO Interface provides a general purpose 16-bit I/O facility, using 16 dedicated device pins. It is organised using two blocks of 8-bit Registers (tabulated below), one block located at Base address 0320h for I/O Ports 0 to 7, the other at Base address 0328h for I/O Ports 8 to 15..

Table 21-1. GPIO Port Registers

Register Name	Mnemonic	Offset Value	Read/Write	Default Value (h)
Port Direction Control Register (Base+00h):	portDirCtrl	Base + 000	read/write	FF
Read Port Control	readPortCtrl	Base + 001	read/write	00
Read Register	readReg	Base + 010	read/write	00
Interrupt Unmask	intrUnMask	Base + 011	read/write	00
Interrupt Edge Select	intrEdgeSele ct	Base + 100	read/write	00
Clear Interrupt	clearIntr	Base + 101	write only	-
GPIO Port	GPIOport	Base + 110	read/write	-
Strap Register	strapReg	Base + 111	read only	Read at Reset from Strap

Each block is an 8-bit slave device which can be integrated onto an ISA-type bus. Each block has the following features:

- Built-in debounce logic for each Input port.
- Each GPIO port is configurable for Rise or Fall edge interrupt generation.
- Strap recording on all 16 ports, 8 ports per block.

Each GPIO port can be configured as an input or an output simply by programming the associated port direction control register. All GPIO ports are configured as inputs at reset, which also latches the input levels into the Strap Registers. The input states of the ports are thus recorded automatically at reset, and this can be used as a strap register anywhere in the system.

All the GPIO ports can be configured to generate rise or fall edge interrupts, but this can only be done when the ports are configured as inputs. These interrupt states can be cleared from the Interrupt Edge Select Register by writing to the Clear Interrupt Register (Base+05h). In cascade mode it is also necessary to clear the interrupt of the Master GPIO.

21.2. GPIO BASE ADDRESS

The following table details the programming for the PCI config space to access GPIO. Note that the GPIOs are disabled at reset

Table 21-2. PCI South Bridge Config Space for GPIO Programming

Offset	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
40	Device Control	Reserved				See GPIO_MDC[1] in Section 8.8.17.	See GPIO_SDC[2] in Section 8.8.17.	See GPIO[3] in Section 8.8.17.	See Section 8.7.6.
44	Device Control	GPIO Master Base Address [7:3]					0	0	1
45	Device Control	GPIO Master Base Address [15:8]							
46	Device Control	GPIO Slave Base Address [7:3]					0	0	1
47	Device Control	GPIO Slave Base Address [15:8]							
48	Device Control	GPIO Debounce Count [7:0]							
49	Device Control	GPIO Debounce Count [15:8]							
50	Device Control	GPIO Debounce Count [23:16]							
51	Device Control	GPIO Debounce Count [31:24]							

Note 1. This is a separate control for Master input port debounce control enable/disable.

Note 2. This is a separate control for Slave input port debounce control enable/disable.

Note 3. Enabling GPIO will enable both master and slave GPIO controllers.

21.2.1. GPIO MASTER BASE ADDRESS

This 16-bit register sets the base addrress of the eight Master GPIO ports.

GPIO_MBA Access = Section Table 21-2. Regoffset = 044h

15	14	13	12	11	10	9	8
GPIO_MBA							
Default value after reset = 0321h							

7	6	5	4	3	2	1	0
GPIO_MBA					0	0	1
Default value after reset = 0321h							

Bit Number	Mnemonic	Description
Bits 15-3	GPIO_MBA	GPIO Master Base Address [157:3]. See Section 21.1.
Bits 2-0	Rsv	These bits are hardwired to 001

21.2.2. GPIO SLAVE BASE ADDRESS

This 16-bit register sets the base addrress of the eight Master GPIO ports.

GPIO_SBA | Access = Section Table 21-2. | Regoffset = 046h

15	14	13	12	11	10	9	8
GPIO_SBA							
Default value after reset = 0329h							

7	6	5	4	3	2	1	0
GPIO_SBA					0	0	1
Default value after reset = 0329h							

Bit Number	**Mnemonic**	**Description**
Bits 15-3	GPIO_SBA	GPIO Master Base Address [15:3]. See Section 21.1.
Bits 2-0	Rsv	These bits are hardwired to 001

21.2.3. GPIO DEBOUNCE COUNT REGISTER

This 32-bit register sets the debounce counter. The debounce counter is disabled at reset.

GPIO_DC Access = Section Table 21-2. Regoffset = 048h

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO_DC															
Default value after reset = 00000000h															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO_DC															
Default value after reset = 00000000h															

Bit Number	Mnemonic	Description
Bits 31-0	GPIO_DC	**GPIO Debounce Count:**

21.3. REGISTER DESCRIPTION

The GPIO requires eight address lines to access all functions. The registers are explained in the following sections.

21.3.1. PORT DIRECTION CONTROL REGISTER (BASE+00H):

This 8-bit register sets the direction, input or output, of each of the eight GPIO ports. After reset this register defaults to 0xFF, setting all ports to the input mode.

portDirCtrl Access = Section 21.1. Regoffset = 000h

7	6	5	4	3	2	1	0
Port 7 (15)	Port 6 (14)	Port 5 (13)	Port 4 (12)	Port 3 (11)	Port 2 (10)	Port 1 (9)	Port 0 (8)
Default value after reset = FFh							

Bit Number	Mnemonic	Description
Bits 7-0		0 = Output, 1= Input.

21.3.2. READ PORT CONTROL REGISTER (BASE+01H):

This is an 8-bit register which decides whether the data read at address Base+06H, for the ports which are configured as outputs, is the data at the GPIO Port or is the data from the read register at address Base+02h.

For all ports which are configured as Inputs, the read data at address Base+06h always returns the data at the GPIO Port.

For any port which is configured as an Output, the following holds true.

A ONE in any bit position returns the read data, at address Base+06h, from the Read Register, which is accessible at address Base+02h.

A ZERO in any bit position returns the read data, at address Base+06h, from the GPIO Port.

readPortCtrl Access = Section 21.1. Regoffset = 001h

7	6	5	4	3	2	1	0
Port 7 (15)	Port 6 (14)	Port 5 (13)	Port 4 (12)	Port 3 (11)	Port 2 (10)	Port 1 (9)	Port 0 (8)
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-0		1: Returns the read data, at address Base+06h, from the Read Register, which is accessible at address Base+02h. 0: Returns the read data, at address Base+06h, from the GPIO Port.

21.3.3. READ REGISTER (BASE+02H):

This is an 8-bit register which is used to return the read data for the ports which are configured as Outputs, provided also that their corresponding bits in the Read Port Control Register are set to 1.

readReg Access = Section 21.1. Regoffset = 010h

7	6	5	4	3	2	1	0
Port 7 (15)	Port 6 (14)	Port 5 (13)	Port 4 (12)	Port 3 (11)	Port 2 (10)	Port 1 (9)	Port 0 (8)
Default value after reset = 00h							

Bit Number	**Mnemonic**	**Description**
Bits 7-0		Read data register for Ports configured as Outputs.

21.3.4. INTERRUPT UNMASK REGISTER (BASE+03H):

This is an 8-bit register with each bit controlling the interrupt mask for the corresponding port. After reset this register is cleared, disabling all interrupt generation. Interrupts can be un-masked only for the ports which are configured as inputs. Clearing the mask for the ports which are configured as outputs has no effect.

intrUnMask Access = Section 21.1. Regoffset = 011h

7	6	5	4	3	2	1	0
Port 7 (15)	Port 6 (14)	Port 5 (13)	Port 4 (12)	Port 3 (11)	Port 2 (10)	Port 1 (9)	Port 0 (8)
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-0		Write interrupt mask bits for selected input ports.

21.3.5. INTERRUPT EDGE REGISTER (BASE+04H):

This is an 8-bit register. Each bit controls the trigger for interrupt generation for the corresponding ports. This has no effect if its corresponding interrupt is masked. After reset this register is cleared, which means that if the port's interrupt is unmasked, then the interrupt trigger condition is the rising edge on the input port. Again, this register is don't care for all ports which are configured as outputs.

.

intrEdgeSelect Access = Section 21.1. Regoffset = 100h

7	6	5	4	3	2	1	0
Port 7 (15)	Port 6 (14)	Port 5 (13)	Port 4 (12)	Port 3 (11)	Port 2 (10)	Port 1 (9)	Port 0 (8)
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-0		Write interrupt edge select bits for selected input ports 0: rising edge 1: falling edge

21.3.6. INTERRUPT CLEAR COMMAND (BASE+05H):

This is a write only address location. A read will return 0FFh.

.

clearIntr Access = Section 21.1. Regoffset = 101h

7	6	5	4	3	2	1	0
Port 7 (15)	Port 6 (14)	Port 5 (13)	Port 4 (12)	Port 3 (11)	Port 2 (10)	Port 1 (9)	Port 0 (8)
Default value after reset = h							

Bit Number	Mnemonic	Description
Bits 7-0		Write 1 to clear interrupt for selected port.

21.3.7. GPIO PORT REGISTER (BASE+06H):

This is an 8-bit register which controls/reads the GPIO port. A write to this with any data will change the value on the ports which are configured as outputs. The ports configured as inputs are not affected. Whereas a read to this address will return the value at the port for those ports which are configured as inputs. For those ports which are configured as outputs, a read will return either the output value OR will return the data from the read register, depending on the programming of the readPortCtrl register.

.

GPIOport Access = Section 21.1. Regoffset = 110h

7	6	5	4	3	2	1	0
Port 7 (15)	Port 6 (14)	Port 5 (13)	Port 4 (12)	Port 3 (11)	Port 2 (10)	Port 1 (9)	Port 0 (8)
Default value after reset = h							

Bit Number	Mnemonic	Description
Bits 7-0		Write output port data, read input port data

21.3.8. STRAP REGISTER (BASE+07H):

This is an 8-bit register and is used to latch the value on the GPIO port at reset. This therefore becomes a strap register and can be used anywhere in the system. This register is read only; a write to this register has no effect.

.

strapReg Access = Section 21.1. Regoffset =111h

7	6	5	4	3	2	1	0
Port 7 (15)	Port 6 (14)	Port 5 (13)	Port 4 (12)	Port 3 (11)	Port 2 (10)	Port 1 (9)	Port 0 (8)
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-0		Read and latch GPIO port state at reset

22. UNIVERSAL SERIAL BUS

22.1. INTRODUCTION

The Universal Serial Bus (USB) is a general-purpose, high-speed, communications interface for connecting peripheral equipment to a PC. The main USB features are listed below:

- Using a special 4-wire cable, the bus can be expanded, using multi-port hubs, to link up to 127 devices to a PC.

- The USB supplies the power to the connected peripheral device, obviating the need for external power supplies.

- The USB port connector is hot-pluggable, i.e. peripherals can be connected or disconnected with power applied to the PC.

- The Plug-and-Play concept is supported, where the PC recognises each peripheral device that is plugged in and loads the appropriate driver.

- Where a new peripheral device has no driver, and cannot run using a generic driver, the user is prompted for a driver to be loaded.

- Simple standard cabling requirements: no null modem cables, no handshaking lines, etc.

- Full speed devices can communicate with the PC at 12 Mbps; keyboards, mice, etc. can communicate at a lower 1.5 Mbps rate to reduce cost.

There are four data transfer types defined in USB. Each type is optimised to match the service requirements between the client software and the USB device. The four types are:

1) Isochronous Transfers: Periodic data transfers with a constant data rate. Data transfers are correlated in time between the sender and the receiver.

2) Control Transfers: Non-periodic data transfers used to communicate configuration/command/status type information between the client software and the USB device.

3) Interrupt Transfers: Small data transfers used to communicate information from the USB device to the client software. The Host Controller Driver polls the USB device by issuing tokens to the device at a periodic interval, sufficient for the requirements of the device.

4) Bulk Transfers: Non-periodic data transfers used to communicate large amounts of information between client software and the USB device.

The USB conforms to the *Open Host Controller Interface (OpenHCI) Specification, Release 1.1,* which provides a register-level description of a Host Controller for the Universal Serial Bus. Refer to this specification document for detailed information.

22.2. OPERATIONAL REGISTERS

The Host Controller (HC) operational registers are mapped into a non-cacheable portion of the system addressable space. These registers, which are used by the Host Controller Driver (HCD), are divided into four partitions:

- Control and Status
- Memory Pointer
- Frame Counter
- Root Hub

As reserved bits may be allocated in future releases, to ensure interoperability, the Host Controller Driver that does not use a reserved field should not assume that the reserved field contains 0. The Host Controller Driver should always preserve the value(s) of the reserved field. When a R/W register is modified, the Host Controller Driver should first read the register, modify the bits desired, then write the register with the reserved bits still containing the read value. Alternatively, the Host Controller Driver can maintain an in-memory copy of previously written values that can be modified and then written to the Host Controller register. When a write to set/clear register is written, bits written to reserved fields should be 0.

Table 22-1. Host Controller Operational Registers

Mnemonic	Offset	Description
Control and Status Partition:		
HcRevision	0	This 8-bit read-only field contains the BCD representation of the version of the HCI specification that is implemented by this HC. For example, a value of 11h corresponds to version 1.1. All of the HC implementations that are compliant with this specification will have a value of 10h.
HcControl	4	This register defines the operating modes for the Host Controller. Apart from **HostControllerFunctionalState** and **RemoteWakeupConnected,** the fields in this register are modified only by the Host Controller Driver
HcCommandStatus	8	The *HcCommandStatus* register is used by the Host Controller to receive commands issued by the Host Controller Driver, as well as reflecting the current status of the Host Controller. To the Host Controller Driver, it appears to be a “write to set” register. The Host Controller must ensure that bits written as ‘1’ become set in the register while bits written as ‘0’ remain unchanged in the register. The Host Controller Driver may issue multiple distinct commands to the Host Controller without concern for corrupting previously issued commands. The Host Controller Driver has normal read access to all bits. The **SchedulingOverrunCount** field indicates the number of frames with which the Host Controller has detected the scheduling overrun error. This occurs when the Periodic list does not complete before EOF. When a scheduling overrun error is detected, the Host Controller increments the counter and sets the **SchedulingOverrun** field in the *HcInterruptStatus* register.

Mnemonic	Offset	Description
HcInterruptStatus	C	This register provides status on various events that cause hardware interrupts. When an event occurs, Host Controller sets the corresponding bit in this register. When a bit becomes set, a hardware interrupt is generated if the interrupt is enabled in the *HcInterruptEnable* register and the **MasterInterruptEnable** bit is set. The Host Controller Driver may clear specific bits in this register by writing ‘1’ to bit positions to be cleared. The Host Controller Driver may not set any of these bits. The Host Controller will never clear the bit.
HcInterruptEnable	10	Each enable bit in the *HcInterruptEnable* register corresponds to an associated interrupt bit in the *HcInterruptStatus* register. The *HcInterruptEnable* register is used to control which events generate a hardware interrupt. When a bit is set in the *HcInterruptStatus* register AND the corresponding bit in the *HcInterruptEnable* register is set AND the **MasterInterruptEnable** bit is set, then a hardware interrupt is requested on the host bus. Writing a '1' to a bit in this register sets the corresponding bit, whereas writing a '0' to a bit in this register leaves the corresponding bit unchanged. On read, the current value of this register is returned.
HcInterruptDisable	14	Each disable bit in the *HcInterruptDisable* register corresponds to an associated interrupt bit in the *HcInterruptStatus* register. The *HcInterruptDisable* register is coupled with the *HcInterruptEnable* register. Thus, writing a '1' to a bit in this register clears the corresponding bit in the *HcInterruptEnable* register, whereas writing a '0' to a bit in this register leaves the corresponding bit in the *HcInterruptEnable* register unchanged. On read, the current value of the *HcInterruptEnable* register is returned.
Memory Pointer Partition:		
HcHCCA	18	The *HcHCCA* register contains the physical address of the Host Controller Communication Area. The Host Controller Driver determines the alignment restrictions by writing all 1s to *HcHCCA* and reading the content of *HcHCC*A. The alignment is evaluated by examining the number of zeroes in the lower order bits. The minimum alignment is 256 bytes; therefore, bits 0 through 7 must always return '0' when read. This area is used to hold the control structures and the Interrupt table that are accessed by both the Host Controller and the Host Controller Driver.
HcPeriodCurrentED	1C	The *HcPeriodCurrentED* register contains the physical address of the current Isochronous or Interrupt Endpoint descriptor.
HcControlHeadED	20	The *HcControlHeadED* register contains the physical address of the first Endpoint Descriptor of the Control list.
HcControlCurrentED	24	The *HcControlCurrentED* register contains the physical address of the current Endpoint Descriptor of the Control list.
HcBulkHeadED	28	The *HcBulkHeadED* register contains the physical address of the first Endpoint Descriptor of the Bulk list.

Mnemonic	Offset	Description
HcBulkCurrentED	2C	The *HcBulkCurrentED* register contains the physical address of the current endpoint of the Bulk list. As the Bulk list will be served in a round-robin fashion, the endpoints will be ordered according to their insertion to the list.
HcDoneHead	30	The *HcDoneHead* register contains the physical address of the last completed Transfer Descriptor that was added to the Done queue. In normal operation, the Host Controller Driver should not need to read this register as its content is periodically written to the HCCA.
Frame Counter Partition:		
HcFmInterval	34	The *HcFmInterval* register contains a 14-bit value which indicates the bit time interval in a Frame, (i.e., between two consecutive SOFs), and a 15-bit value indicating the Full Speed maximum packet size that the Host Controller may transmit or receive without causing scheduling overrun. The Host Controller Driver may carry out minor adjustment on the **FrameInterval** by writing a new value over the present one at each SOF. This provides the programmability necessary for the Host Controller to synchronise with an external clocking resource and to adjust any unknown local clock offset.
HcFmRemaining	38	The *HcFmRemaining* register is a 14-bit down counter showing the bit time remaining in the current Frame.
HcFmNumber	3C	The *HcFmNumber* register is a 16-bit counter. It provides a timing reference among events happening in the Host Controller and the Host Controller Driver. The Host Controller Driver may use the 16-bit value specified in this register and generate a 32-bit frame number without requiring frequent access to the register.
HcPeriodicStart	40	The *HcPeriodicStart* register has a 14-bit programmable value which determines when is the earliest time HC should start processing the periodic list.
HcLSThreshold	44	The *HcLSThreshold* register contains an 11-bit value used by the Host Controller to determine whether to commit to the transfer of a maximum of 8-byte LS packet before EOF. Neither the Host Controller nor the Host Controller Driver are allowed to change this value.
Root Hub Partition:		
HcRhDescriptorA	48	The *HcRhDescriptorA* register is the first register of two describing the characteristics of the Root Hub. Reset values are implementation-specific. The Descriptor length (11), Descriptor type (TBD), and hub controller current (0) fields of the hub Class Descriptor are emulated by the HCD. All other fields are located in the *HcRhDescriptorA* and *HcRhDescriptorB* registers.
HcRhDescriptorB	4C	The *HcRhDescriptorB* register is the second register of two describing the characteristics of the Root Hub. These fields are written during initialization to correspond with the system implementation. Reset values are implementation-specific.

Mnemonic	Offset	Description
HcRhStatus	50	The *HcRhStatus* register is divided into two parts. The lower word of a Dword represents the Hub Statusfield and the upper word represents the Hub StatusChange field. Reserved bits should always be written '0'.
HcRhPortStatus[1]	54	The *HcRhPortStatu*s[1:NDP] register is used to control and report port events on a per-port basis. **NumberDownstreamPorts** represents the number *ofHcRhPortStatus* registers that are implemented in hardware. The lower word is used to reflect the port status, whereas the upper word reflects the status change bits. Some status bits are implemented with special write behaviour (see below). If a transaction (token through handshake) is in progress when a write to change port status occurs, the resulting port status change must be postponed until the transaction completes. Reserved bits should always be written '0'.
...	...	
HcRhPortStatus[NDP]	54+4*NDP	

For a detailed Register description, refer to the *Open Host Controller Interface (OpenHCI) Specification, Release 1.1.*

22.3. PCI CONFIGURATION

22.3.1. PCI INTERFACE

This section describes the configuration registers used to interface with other system components in a PCI-based PC host. Only those bits relevant to the implementation of a USB Host Controller with PCI interface are described here. For the definition of the other bits/registers, refer to the PCI Specification, Revision 2.1.

The registers listed here are accessed for set-up during PCI initialization. They might also be accessed through special cycles during normal system runtime. *Header type 0* is the format for the device's configuration header region, the first 16 Dwords. They are also commonly called the *PCI configuration spaces* of a PCI device. For the USB Host Controller with PCI interface, the operational registers (i.e., *PCI nonconfiguration space*s) that are described in the Operational Registers section (Section 22.2.) are directly memory-mapped into the main memory of the PC host system. "Reset" issued to the Host Controller through its respective programming interface does not affect the contents of the *PCI configuration space* (contents of the operational registers of the Root Hub are also not affected). "Hardware reset" issued by the system logic in the PC host, during system power-up and "cold-boot", causes all of the on-chip registers of the Host Controller and the Root Hub to return their default values.

Note: The LATENCY_TIMER in the *PCI configuration spaces* defines the minimum amount of time that the Host Controller is permitted to retain ownership of the bus after it has acquired bus ownership and has initiated a subsequent transaction. It should be set to a value that reflects the nominal burst size of the underlying device, resulting in a good compromise between the utilization and efficiency of the PCI bus. In determining the value, it should be considered that the maximum size of packet transferred over the USB ranges from 64 bytes to 1023 bytes. A value of '16h' is recommended, as it will allow a total of 24 PCI clocks, sufficient for a burst transfer of 64-byte (assuming a target initial latency of 8 PCI clocks).

Like the other integrated peripherals, the USB Controller uses PCI Interrupt A.

22.3.2. PCI CONFIGURATION SPACES FOR OPENHCI-COMPLIANT USB HOST CONTROLLER

The following table provides a summary of the registers that are necessary for the USB Host Controller to be successfully configured in a PCI-based PC host.

Register	Offset	Description
Command	05 - 04	Provides coarse control over a device’s ability to generate and respond to PCI cycles
CLASS_CODE	0B - 09	Identifies the generic function of the device
BAR_OHCI	13 - 10	Specifies the base address of a contiguous block in the main memory of the PC host, from which 4 KB of directly-mapped addressing spaces are reserved by OpenHCI for the operational registers of the Host Controller

For a detailed Register description, refer to the *Open Host Controller Interface (OpenHCI) Specification, Release 1.1.*

22.3.3. LEGACY SUPPORT REGISTERS

Four operational registers are used to provide the legacy support. Each of these registers is located on a 32-bit boundary. The offset of these registers is relative to the base address of the Host Controller operational registers with *HceControl* located at offset 100h.

Register	Offset	Description
HceControl	100h	Used to enable and control the emulation hardware and report various status information.
HceInput	104h	Emulation side of the legacy Input Buffer register.
HceOutput	108h	Emulation side of the legacy Output Buffer register where keyboard and mouse data is to be written by software.
Hce Status	10Ch	Emulation side of the legacy Status register.

23. SERIAL PORT

23.1. INTRODUCTION

The Serial Port of the STPC is a universal asynchronous receiver/transmitter (UART) which is fully programmable by an 8-bit CPU interface. It supports word lengths from five to eight bits, an optional parity bit and one or two stop bits. If enabled the parity can be odd, even or forced to a defined state. A 16-bit programmable baud rate generator and an 8-bit scratch register are included. Eight modem control lines and a diagnostic loop-back mode are provided. Two 16-Byte FIFOs are included, one for transmit and one for receive. Two DMA handshake lines are provided to indicate when the FIFOs are ready to transfer data to the CPU. An interrupt can be generated from any one of ten sources.

Up to two Serial Ports are available depending on the Strap Options set (see Section 3.). The following bit descriptions relate to one Serial Port, the same bits are also applicable to both Serial Ports, only the register address is different.

23.2. FUNCTIONAL DESCRIPTION

23.2.1. TRANSMIT OPERATION

Transmission is initiated by writing the data to be sent to the Transmitter Holding Register. The data will then be transferred to the Transmit Shift Register together with a start bit and parity and stop bits as determined by the Line Control Register. The bits to be transmitted are then clocked out of the transmit shift register by the transmit clock (BAUD#) which comes from the baud rate generator.

If enabled, an interrupt will be generated when the Transmitter Holding Register becomes empty.

23.2.2. RECEIVE OPERATION

Data is clocked into the receiver by the receive clock (RCLK). The receive clock should be 16 times the baud rate of the received data. A filter is used to remove spurious inputs which last for less than two periods of RCLK. When the complete word has been clocked into the receiver the data bits are transferred to the Receiver Buffer Register to be read by the CPU. The receiver also checks for a stop bit and for correct parity, as determined by the Line Control Register.

If enabled, an interrupt will be generated when the data has been transferred to the Receiver Buffer Register. Interrupts can also be generated for incorrect parity or a missing stop bit (frame error).

23.2.3. MODEM CONTROL LINES

The output modem control lines, RTS# and DTR#, can be set or cleared by writing to the Modem Control Register. The current status of the input modem control line, DCD#, RI#, DSR# and CTS# can be read from the Modem Status Register. Bit 2 of this register will be set if the RI# modem status line has changed from low to high since the register was last read.
If enabled, an interrupt will be generated when DSR#, CTS#, RI# or CD# are asserted.

23.3. SERIAL INTERFACE SIGNALS

SIN1, SIN2, Input Serial input, data is clocked in using RCLK/16.

SOUT1, SOUT2 Output Serial output, data is clocked out using TCLK/16 (TCLK=BAUD#).

DCD1#, DCD2#, Input Data carrier detect, Active low.

RI1#, RI2#, Input Ring indicator, Active low.

DSR1#, DSR2#, Input Data set ready, Active low.

CTS1#, CTS2#, Input Clear to send, Active low.

RTS1#, RTS2#, Output Request to send, Active low.

DTR1#, DTR2#, Output Data terminal ready, Active low.

BAUD# is an internal output transmit timing clock, derived from CLK divided by the value in the divisor latch DLL & DLM.

23.4. REGISTER DESCRIPTION

23.4.1. ADDRESSING

A0-2 and DLAB (Line Control Register bit 7) define which register appears on DA0 - 7. When CE# and WR# are true,

A0-2 and DLAB define which register is to be written with data.

.

Table 23-1. Serial Port Register Addresses

Address	DLAB	Register Name	Comment
000	0	RBR Receiver buffer	Read Only
000	0	THR Transmitter Holding	Write Only
001	0	IER Interrupt Enable	
010	X	IIR Interrupt Ident	Read Only
010	X	FCR FIFO Control	Write Only
011	X	LCR Line Control	
100	X	MCR Modem Control	
101	X	LSR Line Status	Read Only
110	X	MSR Modem Status	Read Only
111	X	SCR Scratch	
000	1	DLL Divisor Latch (LS)	
001	1	DLM Divisor Latch (MS)	

Note: X = don't care, either 0 or 1.

The first Serial Port is addressed as COM1 at IO address 3F8h, the second Serial Port is addressed as COM2 at IO address 2F8h.

23.4.2. RECEIVER BUFFER REGISTER

This is an 8-bit read only register. This register is updated from the receive shift register at the end of a receive sequence.

This register is accessed only when the bit 7 (DLAB) of the Line control register = 0.

RBR Access = 3F8h/2F8h Regoffset = 000h

7	6	5	4	3	2	1	0
Default value after reset = undefined or 00h							

Programming notes:

If the FIFOs are disabled this register is undefined after reset.

If the FIFOs are enabled this register will return zero after a reset if the receive FIFO is empty.

23.4.3. TRANSMITTER HOLDING REGISTER

This is a 8-bit write only register. Data is held in this register until transferred to the transmitter shift register.

This register is accessed only when the bit 7 (DLAB) of the Line control register = 0.

THR Access = 3F8h/2F8h Regoffset = 000h

7	6	5	4	3	2	1	0
Default value after reset =							

23.4.4. INTERRUPT ENABLE REGISTER

This register is accessed only when the bit 7 (DLAB) of the Line control register = 0.

IER Access = 3F8h/2F8h Regoffset = 001h

7	6	5	4	3	2	1	0
Rsv				EDSSI	ELSI	ETBEI	ERBFI
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-4	**Rsv**	**Reserved.** Read as '0'
Bit 3	**EDSSI**	**Enable Modem status interrupt.** When set ("1"), an interrupt is generated if D0, D1, D2 or D3 of the Modem Status Register become set.
Bit 2	**ELSI**	**Enable Rx Status Interrupt.** When set ("1"), an interrupt is generated if D1, D2, D3 or D4 of the Line Status Register become set.
Bit 1	**ETBEI**	**Enable Tx Holding Register Empty Interrupt.** When set ("1"), an interrupt is generated if THRE=1 or the Transmitting Holding Register is empty.
Bit 0	**ERBFI**	**Enable Receiver Buffer Register.** When set ("1"), an interrupt is generated if the Receive Buffer contains data.

23.4.5. INTERRUPT IDENTIFICATION REGISTER

This is a 8-bit read only register.

IIR				Access = 3F8h/2F8h			Regoffset = 002h
7	6	5	4	3	2	1	0
FIFOE		Rsv		ID2	ID1	ID0	INT#
Default value after reset = 01h							

Bit Number	Mnemonic	Description
Bits 7-6	**FIFOE**	**FIFOE.** Returns '1' if FIFOs enabled, otherwise '0'.
Bits 5-4	**Rsv**	**Reserved.** Always returns 0.
Bit 3	**ID2**	**Interrupt ID Bit 2.** If FIFOs disabled returns 0. See Table 23-2.
Bit 2	**ID1**	**Interrupt ID Bit 1.** See Table 23-2.
Bit 1	**ID0**	**Interrupt ID Bit 0.** See Table 23-2.
Bit 0	**INT#**	**Not interrupt pending.** See Table 23-2.

Table 23-2. Interrupt Priority

Bit 3	Bit 2	Bit 1	Bit 0	Priority	Comment
0	0	0	1		No interrupt pending
0	1	1	0	1	Receiver Line Status
0	1	0	0	2	Receive Data Available or RX FIFO trigger
1	1	0	0	2	Character Timeout Indication
0	0	1	0	3	Transmitter Holding Register Empty
0	0	0	0	4	Modem Status

Programming notes:

Pending Interrupts are cleared by the following actions:

Priority 1) Reading line status register,

Priority 2) Reading receive buffer register,

Priority 3) Reading this register if priority 3 interrupt OR writing to the transmitter holding register,

Priority 4) Reading the MODEM status register.

When multiple interrupts are pending the interrupt line pulses low after each service.

23.4.6. RECEIVE TIMEOUT INTERRUPT

A RX FIFO character timeout can be identified when ID2 is ‘1’.

A RX FIFO character timeout occurs if all the following apply:

1. There is at least one character in the FIFO.
2. The most recent character was received longer than four character periods ago (inclusive of all start, parity, and stop bits).
3. The most recent CPU read of the FIFO was longer than four character periods ago.

The character timeout is dependent on the RX clock.

A timeout interrupt is cleared by a CPU read from the RX FIFO.

The timeout timer is restarted on receipt of a new Byte from the input shift register, or on a CPU read from the RX FIFO.

23.4.7. TX FIFO INTERRUPT

The Transmitter Holding Register interrupt occurs when the TX FIFO is empty. It is cleared by writing to the Transmitter Holding Register, or by reading from IIR.

The TX FIFO empty interrupt will be delayed one character period minus the last stop bit period whenever; THRE = 1 and there have not been at least two Bytes in the TX FIFO at the same time since the last time THRE = 1. If the TX interrupt is enabled, setting bit 0 of the FCR will generate an immediate interrupt.

23.4.8. FIFO POLLED OPERATION

If the FIFOs are enabled and at least one of the active bits in IER is disabled, then the Serial Port will operate in the FIFO polled mode. Since the TX and RX paths are controlled separately either one or both can be in the polled mode. The application software should check TX and RX status using the LSR.

23.4.9. FIFO CONTROL REGISTER

This is a 8-bit write only register.

FCR				Access = 3F8h/2F8h			Regoffset = 002h
7	6	5	4	3	2	1	0
RFTL1	RFTL0	Rsv		DMA1	CLRT	CLRR	FIFOE
Default value after reset =							

Bit Number	Mnemonic	Description
Bit 7	**RFTL1**	**RX FIFO trigger level bit 1.** See Table 23-3.
Bit 6	**RFTL0**	**RX FIFO trigger level bit 1.** See Table 23-3.
Bits 5-4	**Rsv**	**Reserved.**
Bit 3	**DMA1**	**Set DMA mode 1.** This bit determines the DMA mode which the TXRDY and RXRDY pins support. On reset, or when this bit is cleared, the device operates in DMA mode 0. When this bit is set the device operates in DMA mode 1. This bit has no effect unless the FIFOE bit is set as well. TXRDY - Mode 0: Goes active (low) when TX FIFO, or TX holding register, is empty. Becomes inactive when a Byte is written to the TX channel. TXRDY - Mode 1: Goes active (low) when there is at least one unfilled position in the FIFO, becomes inactive when the FIFO is full. RXRDY - Mode 0: Becomes active (low) when there is at least one character in the RX FIFO or the holding register is full. It becomes inactive when there are no more characters in the FIFO or holding register. RXRDY - Mode 1: Becomes active (low) when the RX FIFO trigger level or timeout occurs, goes inactive when the RX FIFO is empty.
Bit 2	**CLRT**	**Clear TX FIFO.** Writing a 1 to this bit clears all Bytes in the TX FIFO and resets its counter logic. The output shift register is not affected. This bit is self-clearing.
Bit 1	**CLRR**	**Clear RX FIFO.** Writing a 1 to this bit clears all Bytes in the RX FIFO and resets its counter logic. The input shift register is not affected. This bit is self-clearing.
Bit 0	**FIFOE**	**Enable FIFOs.** Writing a 1 to this bit enables both the RX and TX FIFOs. When the FIFOs are either enabled or disabled, both the RX and the TX FIFOs are reset. This bit must be a 1 for any of the other bits in the register to have any effect.

Table 23-3. RX FIFO Trigger Level Bit 1

Bit 7	Bit 6	RX FIFO Trigger Level (Bytes)
0	0	01
0	1	04
1	0	08
1	1	14

23.4.10. LINE CONTROL REGISTER

LCR Access = 3F8h/2F8h Regoffset = 003h

7	6	5	4	3	2	1	0
DLAB	SB	SP	EPS	PEN	STB	WLS1	WLS0
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**DLAB**	**Divisor Latch Access Bit.** When clear ‘0’, Receive and Transmitter Registers are read/written address 0 and IER register at address 1. When set ‘1’, Divisor Latch LS is read/written at address 0 and Divisor Latch MS read/written at address 1.
Bit 6	**SB**	**Set Break.** When set ‘1’, SOUT signal is forced into the ‘0’ state.
Bit 5	**SP**	**Stick Parity.** When set ‘1’, Parity bit is forced into a defined state, dependent upon state of EPS, PEN: If EPS = ‘1’ & PEN = ‘1’ Parity bit is set and checked = ‘0’. If EPS = ‘0’ & PEN = ‘1’ Parity bit is set and checked = ‘1’.
Bit 4	**EPS**	**Even Parity Select.** When set ‘1’ and PEN = "1" an even number of ones is sent and checked. When clear "0" and PEN = "1" an odd number of ones is sent and checked.
Bit 3	**PEN**	**Parity Enabled.** When set "1" parity is transmitted and checked. Parity bit is added after the data field and before the STOP bits. When clear "0" parity is neither transmitted or checked.
Bit 2	**STB**	**Number of Stop bits.** When set "1" two STOP bits are added after each character is sent, except if character length is 5 then 1½ STOP bits are added. When clear "0" one STOP bit is always added. Only the transmit STOP bits are programmable, the receive stage only expects one STOP bit irrespective of the state of STB.
Bit 1	**WLS1**	**Word Length Select.** Transmitted and Received character size is defined in Table 23-4.
Bit 0	**WLS0**	**Word Length Select.** Transmitted and Received character size is defined in Table 23-4.

Table 23-4. Word Length Select

Bit 1	Bit 0	Character Size
0	0	5 bits
0	1	6 bits
1	0	7 bits
1	1	8 bits

23.4.11. MODEM CONTROL REGISTER

MCR Access = 3F8h/2F8h Regoffset = 004h

7	6	5	4	3	2	1	0
Rsv			Loop	Rsv		RTS	DTR
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-5	**Rsv**	**Reserved.** Read as '0'
Bit 4	**Loop**	**Loop back mode.** When set ‘1’ the following conditions are implemented: 1) SOUT is forced to ‘1’. 2) SIN is disconnected from the Receive input shift register. 3) Receive shift register input is connected to Transmitter shift register output. 4) The Modem status signals are disconnected (CTS#, DSR#, DCD#, RI#). 5) The Modem control signals are connected to modem status inputs (RTS to CTS and DTR to DSR). When clear ‘0’, Modem and control/status signals SIN/SOUT are as normal.
Bits 3-2	**Rsv**	**Reserved.**
Bit 1	**RTS**	**Control Signal.** This signal controls the state of the RTS# output even in loop mode. When RTS = ‘0’ RTS# = ‘1’. When RTS = ‘1’ RTS# = ‘0’.
Bit 0	**DTR**	**Control Signal.** This signal controls the state of the DTR# output even in loop mode. When DTR = ‘0’ DTR# = ‘1’. When DTR = ‘1’ DTR# = ‘0’.

23.4.12. LINE STATUS REGISTER

This is a 8-bit read only register.

LSR Access = 3F8h/2F8h Regoffset = 005h

7	6	5	4	3	2	1	0
FIFOERR	TEMT	THRE	BI	FE	PE	OE	DR
Default value after reset = 60h							

Bit Number	Mnemonic	Description
Bit 7	**FIFOERR**	**RX Data Error in FIFO.** This bit is set to ‘1’ when there is at least one PE, FE, or BI in the RX FIFO. It is cleared by a read from the LSR register, if there are no subsequent errors in the FIFO.
Bit 6	**TEMT**	**Transmitter Empty.** If the FIFOs are disabled, this bit is set to ‘1’ whenever the transmitter holding register and the transmitter shift register are empty. If the FIFOs are enabled, this bit is set whenever the TX FIFO and the transmitter shift register are empty. In both cases this bit is cleared when a Byte is written to the TX data channel.
Bit 5	**THRE**	**Transmitter Holding Register Empty.** If the FIFOs are disabled, this bit is set to ‘1’ whenever the transmitter holding register is empty and ready to accept new data, this bit is cleared when the data is transferred to the transmitter shift register. If the FIFOs are enabled, this bit is set to ‘1’ whenever the TX FIFO is empty. It is cleared when at least one Byte is written to the TX FIFO.
Bit 4	**BI**	**Break Interrupt.** If the FIFOs are disabled, this bit is set whenever the SIN is held in the 0 state for more than a transmission time (START bit + DATA bits + PARITY + STOP bits). BI is reset by the CPU reading this register. If the FIFOs are enabled, this error is associated with the corresponding character in the FIFO. The error is flagged when this Byte is at the top of the FIFO. When break occurs only one zero character is loaded into the FIFO, the next character transfer is enabled when SIN goes into the marking state and receives the next valid start bit.
Bit 3	**FE**	**Framing Error.** If the FIFOs are disabled, this bit is set if the received data did not have a valid STOP bit, FE is reset by the CPU reading this register. If the FIFOs are enabled, the state of this bit is revealed when the Byte it refers to is at the top of the FIFO.
Bit 2	**PE**	**Parity Error.** If the FIFOs are disabled, this bit is set if the received data does not have a valid parity bit, PE is reset by the CPU reading this register. If the FIFOS are enabled, the state of this bit is revealed when the Byte it refers to is at the top of the FIFO.

Bit 1	**OE**	**Overrun Error.** If the FIFOs are disabled, this bit is set if the receive buffer was not read by the CPU before new data from the receive shift register over wrote previous contents. OE is cleared when the CPU reads this register. If the FIFOs are enabled, an overrun error occurs when the RX FIFO is full and the RX shift register becomes full. OE is set as soon as this happens. The character in the shift register is then overwritten, but is not transferred to the FIFO.
Bit 0	**DR**	**Data Ready.** This bit is set whenever the receive buffer is full., or by a Byte being transferred into the FIFO. DR is cleared by the CPU reading the receive buffer, or by reading all of the FIFO Bytes. This bit is also cleared whenever the FIFO enable bit is changed.

23.4.13. MODEM STATUS REGISTER

This is a 8-bit read only register.

MSR Access = 3F8h/2F8h Regoffset = 006h

7	6	5	4	3	2	1	0
DCD	RI	DSR	CTS	DDCD	TERI	DDSR	DCTS
Default value after reset =							

Bit Number	Mnemonic	Description
Bit 7	**DCD**	**Data Carry Detect.** When Loop = ‘0’ this is the complement of input signal DCD#. When Loop = ‘1’ this is equal to OUT2.
Bit 6	**RI**	**Ring Indicator.** When Loop = ‘0’ this is the complement of input signal RI# When Loop = ‘1’ this is equal to OUT1.
Bit 5	**DSR**	**Data Set Ready.** When Loop = ‘0’ this is the complement of input signal DSR#. When Loop = ‘1’ this is equal to DTR.
Bit 4	**CTS**	**Clear To Send.** When Loop = ‘0’ this is the complement of input signal CTS#. When Loop = ‘1’ this is equal to RTS.
Bit 3	**DDCD**	**Delta Data Carry Detect.** This bit is set (‘1’) if the state of DSR has changed since this register was last read.
Bit 2	**TERI**	**Trailing Edge Ring Indicator.** This bit is set if the RI# input changes from ‘0’ to ‘1’ since this register was last read.
Bit 1	**DDSR**	**Delta Data Set Ready.** This bit is set (‘1’) if the state of DSR has changed since this register was last read.
Bit 0	**DCTS**	**Delta Clear to Send.** This bit is set (‘1’) if the state of CTS has changed since this register was last read.

Programming notes:

After reset, Bits 7-4 are inputs, Bits 3-0 = ‘0’ and can be written to.

A modem status interrupt can be cleared by writing '0' or set by writing '1' to this register.Scratch Register

23.4.14. SCRATCH REGISTER

This is a general purpose read/write register.

SCR Access = 3F8h/2F8h Regoffset = 007h

7	6	5	4	3	2	1	0
Default value after reset = undefined							

23.4.15. DIVISOR LATCH (LS) - DIVISOR LATCH (MS)

These registers are accessed only when bit 7 (DLAB) of the Line control register = 1.

DLL Access = 3F8h/2F8h Regoffset = 000h

7	6	5	4	3	2	1	0
Default value after reset = undefined							

DLM Access = 3F8h/2F8h Regoffset = 001h

7	6	5	4	3	2	1	0
Default value after reset = undefined							

Programming notes:

Table 23-5. lists the possible DLL & DLM settings and their respective Baud rates.

Table 23-5. Decimal Divisor

BAUD	DLM Setting	DLL Setting
110	23	82
300	0D	05
1200	03	41
2400	01	A1
4800	00	D0
9600	00	68
19200	00	34
38400	00	1A
57600	00	11

23.5. SPECIAL FEATURES

23.5.1. TRANSMIT MACHINE TIMING

The TXM (Transmit Machine) starts after 2-3 baud clocks from the time the Transmitter Holding Register is written. The SOUT goes low 7-8 baud clocks from the Transmitter Holding Register being written.

23.5.2. THR EMPTY INTERRUPT TIMING

A Transmitter Holding Register Empty interrupt will be generated 17-18 clocks after data has been written to the Transmitter Holding Register, providing that the Transmit Machine was idle when the data was written.

If the Transmitter Holding Register is empty when the Transmitter Holding Register Empty interrupt is enabled an interrupt will be generated immediately.

23.5.3. FIFO RESET TIMING

When using bits 0-3 of the FIFO Control Register to reset the FIFOs the following timing restrictions apply:

FCR0 - Both FIFOs are reset by the master reset (MR), and are held reset unless FCR0 is set to 1.

FCR1 - The RXFIFO clear requires at least one RCLK period to complete the reset and clear itself.

FCR2 - When set to 1, the TXFIFO clear holds the transmit FIFO reset until the leading edge of the next write strobe to the transmit FIFO, or the next read strobe to the IIR.

24. PARALLEL PORT

24.1. INTRODUCTION

The Parallel Port is a multi-function port that transfers information between the host and a peripheral device (e.g. a printer). The parallel port interface contains nine control/status lines and an 8-bit data bus.

The standard PC/AT compatible logical address assignments for LPT1, LPT2 and LPT3 are supported.

The parallel port can be configured for any of the following three modes and supports the IEEE Standard 1284 parallel interface protocol as follows:

– Compatibility Mode (Forward channel, industry-standard parallel port interface ISA),

– Nibble Mode (Reverse channel, compatible with all existing PC hosts),

– Byte Mode (Reverse channel, compatible with IBM PS/2 hosts).

24.2. FUNCTIONAL DESCRIPTION

24.2.1. COMMUNICATION MODES

The interface is initialised in Compatibility Mode. The other modes provide additional features and/or improved performance. Compliant devices can determine and switch to the most effective mode supported by both devices.

The different communication modes are outlined below.

Note: This document covers the operations and functions of the parallel port hardware. The reader should refer to the IEEE standard 1284 for detailed descriptions of the Compatibility, Nibble and Byte protocols. It should also be noted that the hardware operation for Compatibility and Nibble Modes is identical.

24.2.2. COMPATIBILITY MODE

Compatibility Mode provides an asynchronous, byte wide, forward channel (host-to-peripheral), with the data and status lines used according to original definitions, as per the original Centronics port.

24.2.3. NIBBLE MODE

Nibble Mode provides an asynchronous, reverse channel (peripheral-to-host) under the control of the host. Data bytes are transmitted as two sequential, four-bit nibbles using four peripheral-to-host status lines. When the host and/or peripheral do not support bi-directional use of the data lines, Nibble Mode may be used with Compatibility Mode to implement a bi-directional channel. The two modes cannot be active simultaneously.

24.2.4. PS/2 OR BYTE MODE

Byte Mode provides an asynchronous, byte-wide, reverse channel (peripheral-to-host) using the eight data lines of the interface for data and the control/status lines for handshaking. Byte Mode may be used to implement a bi-directional channel, with the transfer direction controlled by the host, when both host and peripheral support bi-directional use of the data lines.

24.2.4.1. Matrix of Protocol Signal Names

Table 24-1. gives the signal names for each protocol.

Table 24-1. Parallel Port Protocol Signal Names

Parallel Port Signal Names	Compatible Signal Names	Nibble Signal Names	Byte Signal Names
PD[7-0]	PD[7-0]		PD[7-0]
SLCT	Select	Xflag	Xflag
ACK#	Ack#	PtrClk	PtrClk
BUSY	Busy	PtrBusy	PtrBusy
PE	pe	AckDataReq	AckDataReq
ERR#	Fault#	DataAvail#	DataAvail#
SLCTIN#	Selectin#	1284 Active	1284 Active
INIT#	INIT#		
STROBE#	Strobe#	HostClk	HostClk
AUTOFD#	AutoFd#	HostBusy	HostBusy

24.3. PARALLEL PORT REGISTERS

24.3.1. PARALLEL PORT CONFIGURATION

The Parallel Port is configured by a set of three programmable registers, accessed through a Configuration Select Register (CSR). These registers are in the default state after power-up and are unaffected by RESET.

24.3.1.1. Configuration Procedure

The following sequence in Table 24-2. is required to program the configuration registers.

Table 24-2. Configuration Register Programming Procedure

Step	Process	Method
1	Enter Configuration.	This requires 55h to be written to port 3F0h (CSR) twice in Mode succession. Note: It is recommended that interrupts be disabled for the duration of the two writes. If a write to another address or port occurs between the two writes, the Parallel Port will not enter Configuration Mode.
2	Configure Registers.	The Parallel Port contains three configuration registers CR1, CR4 and CRA. These registers are accessed by first writing the number of the desired register to port 3F0h (CSR), then writing or reading the selected register through port 3F1h.
3	Exit Configuration Mode.	Configuration Mode is exited by writing AAh to port 3F0h (CSR).

24.3.1.2. Configuration Select Register

This register can only be accessed when the Parallel Port is in Configuration Mode. The CSR is located at port 3F0h and must be initialised upon entering Configuration Mode before the three configuration registers can be accessed, after which it can be used to select which of the configuration registers is to be accessed at port 3F1h.

CSR Access = 3F0h Regoffset =

7	6	5	4	3	2	1	0
Default value after reset =							

24.3.1.3. Configuration Register 1

This register can only be accessed when the Parallel Port is in the Configuration Mode and after CSR has been initialised to 01h.

CR1 Access = 3F1h Regoffset = 001h

7	6	5	4	3	2	1	0
Rsv				PPM	Rsv	PPA	
Default value after power up = 9Fh							

Bit Number	Mnemonic	Description
Bits 7-4	**Rsv**	**Reserved.**
Bit 3	**PPM**	**Parallel Port Mode.** If '1', sets the Parallel Port for Printer Mode (Default). If '0', enables the Extended Parallel Port mode (See Section 24.3.1.4.).
Bit 2	**Rsv**	**Reserved.**
Bits 1-0	**PPA**	**Parallel Port Address.** These bits are used to select the Parallel Port Address (see Table 24-3.).

Table 24-3. Parallel Port Address

Bit 1	Bit 0	Description
0	0	Disabled
0	1	3BCh (LPT3)
1	0	378h (LPT1)
1	1	278h (LPT2)

24.3.1.4. Configuration Register 4

This register can only be accessed when the Parallel Port is in Configuration Mode and after CSR has been initialised to 04h.

CR4			Access = 3F1h				Regoffset = 004h
7	6	5	4	3	2	1	0
Rsv	Rsv					PPEM	
Default value after power up = 00h							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved.** Set/read as '0'.
Bits 6-2	**Rsv**	**Reserved.**
Bits 1-0	**PPEM**	**Parallel Port Extended Modes** (see Table 24-4.).

Table 24-4. Parallel Port Extended Modes

Bit 1	Bit 0	If CR1 (3) = 0 then:
0	0	Standard and Bi-directional Modes(SPP) (Default)

24.3.2. PARALLEL PORT REGISTERS

This section describes Compatibility and Byte Modes.

Each register set description contains the I/O address assignments and a description of the registers and register bits. STAT and CTRL registers are common to all modes.

The base address for the parallel port is determined at power-up. This can be changed by software as described in Section 24.3.1. All registers are accessed as byte quantities.

Some of the registers described contain reserved bits. These have an associated value defined in the register description which is a hard value: it will not change even if these bits are written to. A read from a register that contains reserved bits will return the hard values associated with those bits.

Data is latched into all of the registers on the rising edge of the internal IOW# signal.

24.3.2.1. Compatibility and Byte Modes

The port consists of three registers and can be programmed to operate at three different base addresses: 278h, 378h and 3BCh.

The write locations are:

– (I) Write data to output port (DATA)

    - Base Address + 0h

– (ii) Write command to output port (CTRL)

    - Base Address + 2h

The read locations are:

– (I) Read peripheral data (DATA)

    - Base Address + 0h

– (ii) Read peripheral status data (STAT)

    - Base Address + 1h

– (iii) Read back control register (CTRL)

    - Base Address + 2h

Reads should not be performed from the DATA register in Compatibility mode.

24.3.2.2. Status Register

P_Stat Access = 278h/378h/3BCh Regoffset = 001h

7	6	5	4	3	2	1	0
BUSY	ACK#	PE	SLCT	ERR#	PINTR1	Rsv	Rsv
Default value after power up =							

Bit Number	Mnemonic	Description
Bit 7	**BUSY**	**BUSY.** If asserted, indicates that the peripheral is busy.
Bit 6	**ACK#**	**ACK#.** If asserted, indicates that the peripheral has received a data byte and is ready for another.
Bit 5	**PE**	**PE.** If asserted, indicates that the peripheral is out of paper.
Bit 4	**SLCT**	**SLCT.** If asserted, indicates that the peripheral is selected.
Bit 3	**ERR#**	**ERR#.** If asserted, indicates that the peripheral has a fault.
Bit 2	**PINTR1**	**PINTR1.** This indicates a CPU interrupt by the parallel port, i.e.. the printer has accepted the previous character and is ready for another.
Bit 1	**Rsv**	**Reserved.**
Bit 0	**Rsv**	**Reserved.**

Programming notes:

The content of the STAT register is stored on the falling edge of the internal IOR# signal.

The 'STAT' register is read only, writes to it have no effect.

24.3.2.3. Control Register

P_CTRL Access = 278h/378h/3BCh Regoffset = 002h

7	6	5	4	3	2	1	0
Rsv		PDIB	INTEN	SLCTIN#	INIT#	AUTO	STROBE
Default value after power up = 00h							

Bit Number	Mnemonic	Description
Bits 7-6	**Rsv**	**Reserved.** Always returns '0'.
Bit 5	**PDIB**	**PDIR. Direction bit.** In PS/2 (Byte) Mode, this bit is used to control the direction of the data transfer on the parallel port data bus. Also in Byte Mode, when PDIR = 0 (forward direction), PDOUT[7:0] is enabled; when PDIR = 1 (reverse direction), PDIN[7:0] is enabled.
Bit 4	**INTEN**	**INTEN.** If asserted, allows the peripheral to interrupt the CPU.
Bit 3	**SLCTIN#**	**SLCTIN#.** If asserted, it means the host has selected the peripheral.
Bit 2	**INIT#**	**INIT#.** If asserted, the peripheral is initialised.
Bit 1	**AUTO**	**AUTOFD#.** This tells the printer to advance the paper by one line each time a carriage return is received.
Bit 0	**STROBE**	**STROBE#.** If asserted, this instructs the peripheral to accept the data on the data bus.

25. POWER MANAGEMENT

25.1. INTRODUCTION

For full information on the action of the System Management Mode (SMM), please refer to the STMicroelectronics manual for the ST486 CPU Core. This chapter describes the SMM control registers for the STPC.

The STPC provides the following hardware structures to assist the software in managing system power consumption:

– System Activity detection,
– Three power-down timers,
– Doze timer for detecting short-duration lack of system activity,
– Standby timer for detecting medium-duration lack of system activity,
– Suspend timer for detecting long-term lack of system activity,
– House-keeping activity detection,
– House-keeping timer to cope with short bursts of house-keeping activity while dozing or in standby state,
– Peripheral Activity detection,
– Peripheral timer for detecting lack of peripheral activity,
– STPCLK# modulation to adjust system performance in various system power down states, including full power-on state.

Lack of system activity for progressively longer periods of times is detected by the three power-down timers. These timers can generate a System Management Interrupt (SMI) to the CPU so that the SMM software can put the system in decreasing states of power consumption. System activity in a power-down state can generate an SMI to allow the software to bring the system back up to the full power-on state. The chipset supports up to three power-down states: Doze state, Standby state and Suspend state. These correspond to increasing levels of power savings.

The chipset can detect the presence or absence of the following System activities:

– DMA Request (DRQ),
– Interrupt Request (INTR),
– Parallel IO (PIO),
– Serial IO (SIO) ,
– Keyboard (KBD),
– Floppy Disk Controller (FDC),
– Hard Disk Controller (HDC) ,
– PCI master device,
– Programmable address range.

Each of these can be individually enabled. The presence of an enabled system activity resets the power-down timers. The chipset generates the SMI when no system activity is detected for the delay period programmed in the power-down timers. The software can then put the appropriate sub-systems in the power-down mode, request STPCLK# assertion and stop CPU and other system clocks, program the current power-down state in the chipset and set up the next timer.

The presence of an enabled system activity, when the STPC is in a power-down state, will first enable any stopped clocks, wait for a a programmable delay to allow any internal Phase Locked Loop (PLL) to stabilise and then deassert STPCLK# to enable CPU execution. The device can optionally generate an SMI to allow the SMM to bring the system back to the power-on state.

The current revision of the STPC does not implement support for stopping CPU and other system clocks.

In Doze or Standby state, a house-keeping activity, can bring the system back to full speed for a short period of time before returning back to Doze or Standby state. The chipset can detect the following house-keeping activities:

– DMA Request (DRQ),
– Interrupt Request (INTR),
– Keyboard (KBD),
– PCI master device.

The house-keeping timer determines the length of time the system will be on before returning to the original power-down state. An activity can be either a system activity or a house-keeping activity, but not both at the same time. Further, the Suspend state cannot make use of this feature.

The absence of the following peripheral activities can be enabled to cause an SMI and thus allow the software to put the unused peripherals in the power-down state, while the remainder of the system is still in full power-on state:

– Parallel IO (PIO),
– Serial IO (SIO) ,
– Keyboard (KBD),
– Floppy Disk Controller (FDC),
– Hard Disk Controller (HDC),
– A programmable address range.

Each of these can be individually enabled for inactivity detection. The presence of a peripheral activity does not reset the peripheral timer. It always times out after the programmed delay period. An SMI is generated if any enabled peripheral was not active for this time period. The device provides IO access trapping to detect access to a powered-down peripheral, so that the software can bring the peripheral to the power-on state before the access is completed.

The STPC can also carry out software transparent power management, if so enabled. In this mode of operation, doze and standby time-outs will change the CPU clock without generating an SMI. The state transitions from fully-on to doze or standby and back to fully-on will take place automatically. Also note that the suspend state can never be entered automatically but always requires software assist.

The STPC decodes the various activities listed below in Table 25-1.

Table 25-1. Activity Detected

Activity	Detected via
ISA DMA masters	Low to high transition of hold request of 206
PCI masters	High to low transition of any of PCIRQ2-0#
Parallel port	IO read/write at 378h-37Fh, 278h-27Fh and 3BCh-3BFh
Serial port	IO read/write at 3F8h-3FFh, 2F8h-2FFh, 3E8h-3EFh and 2E8h-2EFh
Keyboard	IO read/write at 60h, 62h, 64h and 66h
Floppy disk	IO read/write at 3F2h, 3F4h, 3F5h and 3F7h
Hard disk	IO read/write in 170h-177h, 376h,1F0h-1F7h and 3F6h address range as well as any bus master activity by the internal IDE controller.

25.2. POWER MANAGEMENT CONTROLLER REGISTERS

25.2.1. TIMER REGISTER 0

This register controls the timer for the selection of the length of timeout for the doze, standby and suspend modes.

Timer0 Access = 0022h/0023h Regoffset = 060h

7	6	5	4	3	2	1	0
SUTT			STT			Rsv	
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-5	**SUTT**	**Suspend Timeout Timer**, when set to any value other than the disable value (000), this timer will generate an SMI on time out. Once enabled, this timer counts down from the programmed valued. If any of the enabled system activities are detected before time out, the timer will reset and start again. These bits are encoded as given in Table 25-2. The suspend timer will count whenever it is not disabled and the suspend time-out bit in the SMI status register 0 is not set to a 1.
Bits 4-2	**STT**	**Standby Timeout Timer,** when set to any value other than the disable value (000) this timer, on expiration, can either generate the SMI to the CPU or if programmed for auto-power saving (software transparent power management) mode, change the power-down state to Standby state (refer to auto-power saving mode for details of the power saving features that are enabled in the standby state). Similar to the Suspend timer, presence of an enabled system activity will reset the timer to restart counting. These bits are encoded as given in Table 25-3. The standby timer will count whenever it is not disabled and the standby time-out bit in the SMI status register 0 is not set to a 1.
Bits 1-0	**Rsv**	**Reserved.**

Table 25-2. Suspend Timer Reset

Bit 7	Bit 6	Bit 5	Suspend Timer reset
0	0	0	Disabled
0	0	1	4 minutes
0	1	0	8 minutes
0	1	1	12 minutes
1	0	0	16 minutes
1	0	1	32 minutes
1	1	0	48 minutes
1	1	1	64 minutes

Table 25-3. Standby Timer Reset

Bit 4	Bit 3	Bit 2	Standby Timer reset
0	0	0	Disabled
0	0	1	Reserved
0	1	0	2 minutes
0	1	1	4 minutes
1	0	0	6 minutes
1	0	1	8 minutes
1	1	0	12 minutes
1	1	1	16 minutes

25.2.2. TIMER REGISTER 1

Timer1 Access = 0022h/0023h Regoffset = 061h

7	6	5	4	3	2	1	0
Rsv	HKT			PTT			Rsv
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	Rsv	**Reserved.**
Bits 6-4	HKT	**House-keeping Timer.** This timer determines how long the PMU will be in Doze house-keeping state when an enabled house-keeping activity is detected while in doze or standby power-down states. It is encoded as given in Table 25-4. The house-keeping counts only when the PMU is in one of the house-keeping states. Another house-keeping activity while the controller is in house_keeping state will reset the house-keeping timer to restart counting. A system activity detection in the house_keeping state will have the same effect as if the controller was in Doze or Standby state. Either an SMI will be generated to allow the software to bring the system to power-on state or the controller will automatically transition to power-on state. The house-keeping timer and function can be disabled by masking out all activity detection via the House-keeping Enable registers.
Bits 3-1	PTT	**Peripheral Timeout Timer.** When set to a value other than (000) this timer on expiration, will generate an SMI if any of the enabled peripherals remained inactive during the entire period. Unlike the power-down timers, the peripheral timer does not reset due to an enabled peripheral activity. It always times out after the programmed delay. An SMI is generated only if any of the enabled peripherals were inactive during this period. This field is encoded as given in Table 25-5. The peripheral timer counts whenever it is enabled.
Bit 0	Rsv	**Reserved.**

Table 25-4. House-keeping Timer Reset

Bit 6	Bit 5	Bit 4	House-keeping Timer reset
0	0	0	Disabled
0	0	1	64 micro-seconds
0	1	0	128 micro-seconds
0	1	1	256 micro-seconds
1	0	0	Reserved
1	0	1	4 milli-seconds
1	1	0	16 milli-seconds
1	1	1	32 milli-seconds

Table 25-5. Peripheral Timer Reset

Bit 3	Bit 2	Bit 1	Peripheral Timer reset
0	0	0	Disabled
0	0	1	8 seconds
0	1	0	16 seconds
0	1	1	32 seconds
1	0	0	64 seconds
1	0	1	128 seconds
1	1	0	256 seconds
1	1	1	512 seconds

25.2.3. TIMER REGISTER 2

Timer 2 Access = 0022h/0023h Regoffset = 08Dh

7	6	5	4	3	2	1	0
DTT			Rsv				
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7- 5	**DTT**	**Doze Timeout Timer.** When set to any value other than the disable value (00), this timer, on expiration, can either generate the SMI to the CPU or if programmed for auto-power saving (software transparent power management) mode, change the power-down state to Doze state (refer to auto-power saving mode for details of the power saving features that are enabled with Doze state). Similar to the suspend timer, presence of an enabled system activity will reset the timer to restart counting. This 3-bit field is encoded as given in Table 25-6. The doze timer will count whenever it is not disabled and the doze time-out bit in the SMI status register 0 is not set to a ‘1’.
Bits 4-2	**Rsv**	**Reserved.**

Table 25-6. Doze Timer Reset

Bit 7	Bit 6	Bit 5	Doze Timer reset
0	0	0	Disabled
0	0	1	50 milli-seconds
0	1	0	100 milli-seconds
0	1	1	500 milli-seconds
1	0	0	Reserved
1	0	1	4 seconds
1	1	0	8 seconds
1	1	1	16 seconds

25.2.4. SYSTEM ACTIVITY ENABLE REGISTER 0

This is the first of the three registers that control which system activity to detect.

Sys_Activ_en0 Access = 0022h/0023h Regoffset = 062h

7	6	5	4	3	2	1	0
DRQ	PCIM	PIO	SIO	KBD	FDC	HDC	Rsv
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**DRQ**	**DMA Request (DRQ).**
Bit 6	**PCIM**	**PCI master device (PCIM).**
Bit 5	**PIO**	**Parallel IO (PIO).**
Bit 4	**SIO**	**Serial IO (SIO).**
Bit 3	**KBD**	**Keyboard (KBD).**
Bit 2	**FDC**	**Floppy Disk Controller (FDC).**
Bit 1	**HDC**	**Hard Disk Controller (HDC).**
Bit 0	**Rsv**	**Reserved.**

Programming notes:

When detected, the power-down timers will reload with their initial time values, or if enabled via the SMI control register, an SMI will be generated, or if programmed for auto-power down mode and in Doze or Standby power-down states, transition to power-on state will take place. Set the following bits to ‘1’ to detect the associated activity, and to ‘0’ to ignore the associated activity.

25.2.5. SYSTEM ACTIVITY ENABLE REGISTER 1

This is the second of the three registers that control which system activity to detect.

Sys_Activ_en1 Access = 0022h/0023h Regoffset = 063h

7	6	5	4	3	2	1	0
Rsv		AR0	Rsv				
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-6	**Rsv**	**Reserved.** Must be programmed to ‘0’.
Bit 5	**AR0**	**Address range 0.**
Bits 4-0	**Rsv**	**Reserved.** Must be programmed to ‘0’.

25.2.6. SYSTEM ACTIVITY ENABLE REGISTER 2

This is the third of the three registers that control which system activity to detect.

Sys_Activ_en2 Access = 0022h/0023h Regoffset = 064h

7	6	5	4	3	2	1	0
IRQ15-1	IRQ0	NMI	Rsv				
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**IRQ15-1**	**IRQ15-1** detection enabled.
Bit 6	**IRQ0**	**IRQ0** detection enabled.
Bit 5	**NMI**	**NMI** detection enable.
Bits 4-0	**Rsv**	**Reserved.**

25.2.7. HOUSE-KEEPING ACTIVITY ENABLE REGISTER 0

This register controls which house-keeping activity to detect. House-keeping activities are detected only in Doze and Standby states. If enabled, a house-keeping activity reverts the system back to power-on state for a short period of time, programmed in the house-keeping timer. Set the following bits to a ‘1’ to enable activity detection and a ‘0’ to ignore the associated activity.

HK_Activ_en0 Access = 0022h/0023h Regoffset = 065h

7	6	5	4	3	2	1	0
DRQ	PCI MD	KBD	IRQ15-1	IRQ0	NMI	Rsv	
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**DRQ**	**DMA Request (DRQ)** activity
Bit 6	**PCI MD**	**PCI master** device activity
Bit 5	**KBD**	**Keyboards** (KBD) activity
Bit 4	**IRQ15-1**	**IRQ15-1** activity
Bit 3	**IRQ0**	**IRQ0** activity
Bit 2	**NMI**	**NMI** activity
Bits 1-0	**Rsv**	**Reserved.**

25.2.8. HOUSE-KEEPING ACTIVITY ENABLE REGISTER 1

This is the second house-keeping activity detection enable register.

HK_Activ_en1 Access = 0022h/0023h Regoffset = 066h

7	6	5	4	3	2	1	0
Rsv		AR0	Rsv				
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-6	**Rsv**	**Reserved.** Must be programmed to ‘0’.
Bit 5	**AR0**	**Address range 0**.
Bits 4-0	**Rsv**	**Reserved.** Must be programmed to ‘0’.

25.2.9. PERIPHERAL INACTIVITY DETECTION REGISTER 0

This register controls which peripheral inactivity is enabled for generating an SMI on a peripheral time-out.

Perif_Inact0 Access = 0022h/0023h Regoffset = 067h

7	6	5	4	3	2	1	0
PIO	SIO	KBD	FDC	HDC	ARO	Rsv	
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**PIO**	**Parallel IO (PIO)** activity.
Bit 6	**SIO**	**Serial IO (SIO)** activity.
Bit 5	**KBD**	**Keyboard (KBD)** activity.
Bit 4	**FDC**	**Floppy Disk Controller (FDC)** activity.
Bit 3	**HDC**	**Hard Disk Controller (HDC)** activity.
Bit 2	**AR0**	**Address range 0.**
Bits 1-0	**Rsv**	**Reserved.** Must be programmed to ‘0’.

Programming notes:

Lack of peripheral activity for an enabled peripheral for one peripheral time-out period generates an SMI. A ‘1’ in a bit position enables the SMI generation for the associated peripheral and a ‘0’ disables it. Software can use the Peripheral Inactivity status register to determine which peripheral should be powered down.

25.2.10. PERIPHERAL ACTIVITY DETECTION REGISTER 0

This register controls which peripheral accesses will cause an SMI.

Perif_Act0 Access = 0022h/0023h Regoffset = 069h

7	6	5	4	3	2	1	0
PIO	SIO	KBD	FDC	HDC	ARO	Rsv	
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**PIO**	**Parallel port (PIO)** access
Bit 6	**SIO**	**Serial port (SIO)** access
Bit 5	**KBD**	**Keyboard (KBD)** access
Bit 4	**FDC**	**Floppy Disk Controller (FDC)** access
Bit 3	**HDC**	**Hard Disk Controller (HDC)** access
Bit 2	**AR0**	**Address range 0**
Bits 1-0	**Rsv**	**Reserved.** Must be programmed to ‘0’

Programming notes:

Typically the power management software will detect non-usage of a peripheral device via Peripheral inactivity status registers, bring the peripheral into power down state and then enable trapping access to that peripheral via this register.

Thus when an application attempts to make use of a powered down peripheral, the access is trapped and an SMI is generated to allow software to re-power the peripheral device before allowing the access to complete. This register is the first of the two such registers.

A ‘1’ in a bit position enables SMI generation for the associate peripheral and a ‘0’ disables.

25.2.11. PERIPHERAL ACTIVITY DETECTION REGISTER 1

This is the second register that controls which peripheral accesses will cause an SMI. This register is similar in functionality to Peripheral Activity detection register 0.

Perif_Act1 Access = 0022h/0023h Regoffset = 06Ah

7	6	5	4	3	2	1	0
Rsv							
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-6	**Rsv**	**Reserved.** Must be programmed to ‘0’.

25.2.12. ADDRESS RANGE 0 REGISTER 0

This register contains bits which are compared with PCI address bits 31-24 if range compare is enabled for memory cycle or compared against bits 15-8 if range compare is enabled for IO cycles.

Add_Rang0-0 Access = 0022h/0023h Regoffset = 06Bh

7	6	5	4	3	2	1	0
Default value after reset = 00h							

25.2.13. ADDRESS RANGE 0 REGISTER 1

Add_Rang0-1 Access = 0022h/0023h Regoffset = 06Ch

7	6	5	4	3	2	1	0
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-3		These bits are compared with PCI address bits 23-19 if range compare is enabled for memory cycle, or compared against bits 7-3 if range compare is enabled for IO cycles.
Bit 2		This bit is compared with PCI address bit 18 if range compare is enabled for memory cycle, or compared with address bit 2 if range compare is enabled for IO cycles and range is 4-Bytes. Otherwise this bit when 1 specifies that the range of IO address to be compared is 16-Bytes and when 0, the range is 8-Bytes.
Bit 1		This bit is compared with PCI address bit 17 if range compare is enabled for memory cycle. Otherwise if range compare is enabled for IO cycles, this bit if 1 specifies that the range of IO address to be compared is 8/16-Bytes and when 0 the range is 4-Bytes.
Bit 0		This bit when ‘1’ specifies that range compare should be done for memory cycles and when ‘0’, for IO cycles.

25.2.14. SMI CONTROL REGISTER 0

This register controls the generation of an SMI, as follows:

SMI_Cont0 Access = 0022h/0023h Regoffset = 071h

7	6	5	4	3	2	1	0
							Rsv
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7		If ‘1’ then generate SMI on Doze time-out. Otherwise if set to a ‘0’, the hardware will transition to Doze state automatically on Doze time-out.
Bit 6		If ‘1’ then generate SMI on Standby time-out. Otherwise if set to a ‘0’, the hardware will transition to Standby state automatically on Standby time-out.
Bit 5		If ‘1’ then generate SMI on Suspend time-out. Otherwise if set to a ‘0’, SMI is not generated. The hardware never transitions into Suspend state by itself.
Bit 4		If ‘1’ then generate SMI on House-keeping time-out. Otherwise if set to a ‘0’, the hardware will automatically transition back to the doze or standby state (which ever state it was in before entering house-keeping state).
Bit 3		If ‘1’ then generate SMI on detecting a house-keeping activity. Otherwise if set to a ‘0’, and if in Doze or Standby state, the hardware will automatically transition to the associated house_keeping states for the duration programmed in the house-keeping timer.
Bit 2		If ‘1’ then generate SMI on detecting a system activity. Otherwise if set to a ‘0’, and if in Doze or Standby state, the hardware will automatically transition to Power-on state on detecting a unmasked system activity. This bit will typically be set to a 1’ by software on entering a power-down state so that a system activity can wake up the system.
Bit 1		This is a write only bit. Setting this bit to a ‘1’ sets bit-7 of the SMI status register 1 and generates an SMI. This bit however will always read back as ‘0’.
Bit 0	**Rsv**	**Reserved.**

25.2.15. SMI STATUS REGISTER 0

This register contains the status information pertaining to the SMI.

SMI_Stat0 Access = 0022h/0023h Regoffset = 073h

7	6	5	4	3	2	1	0
DTO	STO	STO	HKT	HKA	SAD	PID	PAD
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**DTO**	**Doze time-out.** This bit is set to a ‘1’ when Doze time-out occurs. An SMI will be generated if associated SMI enable bit in SMI Control register is set to a ‘1’. The software must write this bit to a ‘1’ to deassert SMI#. If SMI generation has been disabled then the controller will automatically transition to Doze state. This bit will then be cleared on transition from Doze or Standby to Power-on state.
Bit 6	**STO**	**Standby time-out.** This bit will be set to a ‘1’ when Standby time-out occurs. An SMI will be generated if the associated SMI enable bit in SMI Control register is set to a ‘1’. The software must write this bit to a ‘1’ to deassert SMI#. If SMI generation has been disabled then the hardware will automatically transition to Standby state. This bit will then be cleared on transition Standby to Power-on state.
Bit 5	**STO**	**Suspend time-out.** This bit will be set to a ‘1’ when Suspend time-out occurs. An SMI will be generated if the associated SMI enable bit in the SMI Control register is set to a ‘1’. The software must write this bit to a ‘1’ to deassert SMI#.
Bit 4	**HKT**	**House-keeping timeout detected.** This bit will be set to a ‘1’ if the controller is in one of the house-keeping states and the house-keeping timer expires. An SMI will be generated if the associated SMI enable bit in the SMI Control register is set to a ‘1’. The software must write this bit to a ‘1’ to deassert SMI#. If the SMI generation has been disabled, the hardware will automatically transition to doze or standby state. This bit then will be cleared on transition from Doze or Standby states to any other state.
Bit 3	**HKA**	**House-keeping activity detected.** This is a read-only bit and represents the OR of the System activity status registers masked (ANDed) with the corresponding bits in the House-keeping Activity enable registers. An SMI will be generated when this bit is a 1’ and if the associated SMI enable bit in the SMI Control register is set to a ‘1’. The software can refer to Activity status register to determine the cause of the interrupt. The software must clear the corresponding bits of the Activity Status register to deassert SMI#. If SMI generation has been disabled and if the controller in Doze or Standby state, it will automatically transition to House-keeping state.

Bit Number	Mnemonic	Description
Bit 2	**SAD**	**System Activity detected**. This is a read-only bit and represents the OR of the System activity Status registers masked (ANDed) with the corresponding bits of the System Activity enable registers. An SMI will be generated if this bit is a ‘1’ and if the associated SMI enable bit in SMI Control register is set to a ‘1’. The software can refer to Activity status register to determine the cause of this interrupt. The software must clear the System Activity Status registers bits for the enabled system activities to deassert SMI#. If SMI generation has been disabled and if the controller is in Doze or Standby state, it will automatically transition to Power-on state.
Bit 1	**PID**	**Peripheral Inactivity detected.** This is a read-only bit and represents the OR of Peripheral Inactivity Status register bits masked (ANDed) with the associated Peripheral inactivity detection register bit. An SMI# will be generated when this bit is a ‘1’. The software can refer to Peripheral Inactivity status registers to determine which peripheral should be powered down. The software must clear the corresponding bits of the Peripheral Inactivity detection register to deassert SMI#.
Bit 0	**PAD**	**Peripheral Activity Detected.** This is a read-only bit and represents the OR of the System activity Status register masked (ANDed) with the corresponding bits of the Peripheral Activity detection registers. An SMI will be generated when this bit is a ‘1’. The software can refer to the System Activity status register to determine which peripheral caused the interrupt. The software must clear the corresponding bits of the System activity register to deassert SMI#.

Programming notes:

The SMI# output is a logical OR of all the bits (ANDed with their respective SMI generation enable bits) in this register. SMI# output will be deasserted within 3 PCI clocks after the cause of the SMI# is cleared. This register defaults to 00h after reset deasserting SMI# output.

25.2.16. SMI STATUS REGISTER 1

This register is similar to SMI Status register 0 in that it reports the cause of the SMI to the software.

SMI_Stat1 Access = 0022h/0023h Regoffset = 074h

7	6	5	4	3	2	1	0
S SMI	Rsv						
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**S SMI**	**Software SMI.** This bit is set to a ‘1’ by write writing a ‘1’ in bit-1 of the SMI Control register. The software must clear this bit to deassert SMI#.
Bits 6-0	**Rsv**	**Reserved.**

25.2.17. PERIPHERAL INACTIVITY STATUS REGISTER 0

This register contains a ‘1’ in a bit position if the associated peripheral was inactive for the entire duration of the last peripheral time-out period.

Perif_Stat0 Access = 0022h/0023h Regoffset = 075h

7	6	5	4	3	2	1	0
PIO	SIO	KBD	FDC	HDC	ARO	Rsv	
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**PIO**	**Parallel port (PIO)** activity.
Bit 6	**SIO**	**Serial port (SIO)** activity.
Bit 5	**KBD**	**Keyboard (KBD)** activity.
Bit 4	**FDC**	**Floppy Disk Controller (FDC)** activity.
Bit 3	**HDC**	**Hard Disk Controller (HDC)** activity.
Bit 2	**AR0**	**Address range 0**
Bits 1-0	**Rsv**	**Reserved.**

It can also be cleared by software by writing a ‘1’ in the bit which is set to ‘1’.

Programming notes:

A bit in this register is set to a ‘1’ only at peripheral timer time-out. It is set to a ‘0’ as soon as an activity from the associated peripheral is detected.

The status reflected in this register is not conditioned by whether or not the peripheral was enabled for inactivity detection through the Peripheral Inactivity Detection registers. The SMI however will be generated only if any of the enabled peripherals (via Peripheral Inactivity Enable register) were inactive for the entire duration of the peripheral time out.

25.2.18. ACTIVITY STATUS REGISTER 0

This register records presence of activity.

Activ_Stat0 Access = 0022h/0023h Regoffset = 077h

7	6	5	4	3	2	1	0
DRQ	PCIM	PIO	SIO	KBD	FDC	HDC	Rsv
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**DRQ**	**DMA Request (DRQ)** activity.
Bit 6	**PCIM**	**PCI master device (PCIM)** activity.
Bit 5	**PIO**	**Parallel IO (PIO)** activity.
Bit 4	**SIO**	**Serial IO (SIO)** activity.
Bit 3	**KBD**	**Keyboard (KBD)** activity.
Bit 2	**FDC**	**Floppy Disk Controller (FDC)** activity.
Bit 1	**HDC**	**Hard Disk Controller (HDC)** activity.
Bit 0	**Rsv**	**Reserved.**

Programming notes:

A ‘1’ in a bit position indicates that presence of the associated activity since the bit was last cleared. Once set, a bit of this register can only be cleared by software writing a ‘1’ to it or by reset or if auto power management is enabled then any transition to Doze or Standby state (including the ones from house-keeping states) will clear all enabled System and House-keeping activities.

The status reflected in this register is not conditioned by the settings of System Activity Enable, House-keeping Activity Enable, Peripheral Inactivity or Peripheral Activity Detection registers.

25.2.19. ACTIVITY STATUS REGISTER 1

This register is similar to Activity Status register 0. It contains the status for the following bits.

Activ_Stat1 Access = 0022h/0023h Regoffset = 078h

7	6	5	4	3	2	1	0
Rsv		AR0	Rsv				
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-6	**Rsv**	**Reserved.** Must be programmed to ‘0’
Bit 5	**AR0**	**Address range 0**
Bits 4-0	**Rsv**	**Reserved.** Must be programmed to ‘0’

25.2.20. ACTIVITY STATUS REGISTER 2

This register is similar to Activity Status registers 0 and 1.

Activiv_Stat2 Access = 0022h/0023h Regoffset = 079h

7	6	5	4	3	2	1	0
IRQ15-1	IRQ0	NMI	Rsv				
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**IRQ15-1**	**IRQ15-1** activity.
Bit 6	**IRQ0**	**IRQ0** activity.
Bit 5	**NMI**	**NMI** activity.
Bits 4-0	**Rsv**	**Reserved.**

25.2.21. PMU STATUS REGISTER

This register contains the state the power management controller currently is in.

PMU Access = 0022h/0023h Regoffset = 07Ah

7	6	5	4	3	2	1	0
Rsv	PMU	PMU	PMU	PMU	PMU	PMU	PMU
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved.**
Bit 6	**PMU**	**PMU microsecond clock test mode.** This bit is for factory use only and must be set to a ‘0’ by the software. Setting this bit to a 1’ causes the micro-second clock to tick at oscillator clock frequency instead of every microsec-ond.
Bit 5	**PMU**	**PMU millisecond clock test mode.** This bit is for factory use only and must be set to a ‘0’ by the software. Setting this bit to a 1’ causes the millisecond clock to tick at oscillator clock frequency instead of every millisecond.
Bit 4	**PMU**	**PMU second clock test mode.** This bit is for factory use only and must be set to a ‘0’ by the software. Setting this bit to a 1’ causes the second clock to tick at oscillator clock frequency instead of every second.
Bit 3	**PMU**	**PMU minute clock test mode.** This bit is for factory use only and must be set to a ‘0’ by the software. Setting this bit to a 1’ causes the minute clock to tick at oscillator clock frequency instead of every minute.
Bit 2	**PMU**	**PMU state** (see Table 25-7).
Bit 1	**PMU**	**PMU state** (see Table 25-7).
Bit 0	**PMU**	**PMU state** (see Table 25-7).

Table 25-7. PMU State

Bit 2	Bit 1	Bit 0	PMU state
0	0	0	Power-on
0	0	1	Doze
0	1	0	Standby
0	1	1	Suspend
1	0	1	Doze_house_keeping
1	1	0	Standby_house_keeping
1	1	1	Reserved

The architecture allows for: (1) the software to explicitly program the power-down state of the controller, or (2) the controller can change states automatically (auto-power down mode of operation), or (3) a mix of the two. Some power-down states are entered and exited automatically by the hardware, while others require software assist. This is based on the SMI Control register settings as follows:

Transition from Power-on to Doze state will take place automatically on Doze time-out, if bit-7 of the SMI control register is set to a ‘0’. Otherwise if bit-7 is programmed to be a ‘1’, an SMI will be generated and the software can change the state to Doze.

Transition from Doze to Power-on will take place automatically in the presence of an enabled system activity if bit-2 of the SMI control register is programmed to ’0’. Otherwise if bit-2 is programmed to a ‘1’, an SMI will be generated and software can change the state to Power-on.

Transition from Doze to Doze_house_keeping state will take place automatically if an enabled house_keeping activity is detected and bit-4 of the SMI control register is set to a ‘0. Otherwise if bit-7 is programmed to be a ‘1’, an SMI will be generated.

Transition from Doze_house_keeping state to Doze will take place automatically on house-keeping time-out if bit-3 of the SMI control register is set to a ‘0’. Otherwise an SMI will be generated.

Transition from Doze_house_keeping state to Power-on will take place on detecting an enabled system activity automatically if bit-2 of the SMI control register is programmed to ‘0’. Otherwise an SMI will be generated.

Transitions from Doze or Power-on state to Standby will take place automatically on standby time-out if bit-6 of the SMI control register is set to a ‘0’. Otherwise an SMI will be generated.

Transition from Standby to Power-on will take place automatically in presence of an enabled system activity if bit-2 of the SMI control register is programmed to be a ‘0’. Otherwise if bit-2 is programmed to a ‘1’, an SMI will be generated and software can change the state to Power-on.

Transition from Standby to Standby_house_ keeping state will take place automatically if an enabled house_keeping activity is detected and bit-4 of the SMI control register is set to a ‘0’. Otherwise if bit-7 is programmed to be a ‘1’, an SMI will be generated.

Transition from Standby_house_keeping state to Standby will take place automatically on house-keeping time-out if bit-3 of the SMI control register is set to a ‘0’. Otherwise an SMI will be generated.

Transition from Standby_house_keeping state to Power-on will take place on detecting an enabled system activity automatically if bit-2 of the SMI control register is programmed to ‘0’. Otherwise SMI interrupt will be generated.

The hardware never transitions to Suspend state automatically.

The power saving features associated with each power-down state are independent of how the state was entered.

25.2.22. GENERAL PURPOSE REGISTER

This is a read/write IO register that can be used by software.

GP Access = 0022h/0023h Regoffset = 07Bh

7	6	5	4	3	2	1	0
GP	GP	GP	GP	GP	GP	GP	GP
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bit 7	**GP**	**General Purpose Register Bit 7.**
Bit 6	**GP**	**General Purpose Register Bit 6.**
Bit 5	**GP**	**General Purpose Register Bit 5.**
Bit 4	**GP**	**General Purpose Register Bit 4.**
Bit 3	**GP**	**General Purpose Register Bit 3.**
Bit 2	**GP**	**General Purpose Register Bit 2.**
Bit 1	**GP**	**General Purpose Register Bit 1.**
Bit 0	**GP**	**General Purpose Register Bit 0.**

Programming notes:

Writing to this register also updates the external ‘373 latch that can be used to control external devices for power-down purposes. Reads of this register return the value of this internal register.

The GPIOCS# signal will be asserted when writing to this register to latch the data on the ISA data bus.

25.2.23. CLOCK CONTROL REGISTER 0

This register allows control over power saving via stop clock modulation. The power-saving can be tuned to the power-management state the PMU is in.

Clk_Cont0 Access = 0022h/0023h Regoffset = 07Ch

7	6	5	4	3	2	1	0
STPCLK			DSSS			STPCLK	Rsv
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-5	**STPCLK**	**Power-on and housekeeping states** STPCLK# modulation control. These bits control the duty cycle of STPCLK# deassertion when the PMU is in Power-on or one of the house-keeping states given in Table 25-8. The STPCLK# is deasserted and the duty-cycle control ignored if an SMI is pending.
Bit 4-2	**DSSS**	**Doze/Standby/Suspend states** STPCLK# modulation control. These bits control the duty cycle of the STPCLK# deassertion when PMU is in one of the power-down states as given in Table 25-9. The STPCLK# is deasserted and the duty-cycle control ignored if an SMI is pending.
Bit 1	**STPCLK**	**STPCLK# modulation period.** If '1' then the period is 64ms else, if '0', then the period is 64µs.
Bit 0	**Rsv**	**Reserved.**

Table 25-8. Power-on and Housekeeping States

Bit 7	Bit 6	Bit 5	Ratio	Power-on STPCLK# Modulation
0	0	0	1	STPCLK# is never asserted
0	0	1	1/2	1 half period
0	1	0	1/4	1 quarter period
0	1	1	1/8	one-eighth period
1	0	0	1/16	one-sixteenth period
1	0	1	1/32	1/32 period
1	1	0	1/64	1/64 period
1	1	1		*Reserved.*

Table 25-9. Doze/Standby/Suspend States

Bit 4	Bit 3	Bit 2	Ratio	Doze STPCLK# Modulation
0	0	0	1	STPCLK is never asserted
0	0	1	1/2	1 half period
0	1	0	1/4	1 quarter period
0	1	1	1/8	one-eighth period
1	0	0	1/16	one-sixteenth period
1	0	1	1/32	1/32 period
1	1	0	1/64	1/64 period
1	1	1	0	The entire period

25.2.24. DOZE TIMER READ BACK REGISTER

This read only register is provided for test purposes to read back the current value of the upper 8-bits of the 9-bit doze timer.

Doze Access = 0022h/0023h Regoffset = 088h

7	6	5	4	3	2	1	0
Default value after reset = 00h							

Bit Number	Mnemonic	Description
Bits 7-0		Bits 8-1 of the current value of the doze timer.

Programming notes:

This register should not be used by the software.

Note that bit 0 of the current value of the doze timer is not readable.

25.2.25. STANDBY TIMER READ BACK REGISTER

This read only register is provided for test purposes to read back the current value of 5-bit standby timer.

Standby Access = 0022h/0023h Regoffset = 089h

7	6	5	4	3	2	1	0
Rsv							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-5	**Rsv**	**Reserved**.
Bits 4-0		Bits 4-0 of the current value of the standby timer.

Programming notes:

This register should not be used by software.

25.2.26. SUSPEND TIMER READ BACK REGISTER

This read only register is provided for test purposes to read back the current value of the 7-bit Suspend timer.

Suspend Access = 0022h/0023h Regoffset = 08Ah

7	6	5	4	3	2	1	0
Rsv							
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bit 7	**Rsv**	**Reserved**.
Bits 6-0		Bits 6-0 of the current value of the suspend timer.

Programming notes:

This register should not be used by software.

25.2.27. HOUSE-KEEPING TIMER READ BACK REGISTER

This read only register is provided for test purposes to read back the current value of the upper 8-bits of the 9-bit house-keeping timer.

HK_Timer Access = 0022h/0023h Regoffset = 08Bh

7	6	5	4	3	2	1	0
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0		Bits 8-1 of the house-keeping timer.

Programming notes:

This register should not be used by software.

25.2.28. PERIPHERAL TIMER READ BACK REGISTER

This read only register is provided for test purposes to read back the current value of the upper 8-bits of the 9-bit Peripheral timer.

Perif_Timer Access = 0022h/0023h Regoffset = 08Ch

7	6	5	4	3	2	1	0
Default value after reset = undefined							

Bit Number	Mnemonic	Description
Bits 7-0		Bits 8-1 of the Peripheral timer.

Programming notes:

This register should not be used by software.